MASTER EN SCIENCES INFORMATIQUES

Étude des techniques de recouvrement de fichiers dans un système d'ordinateurs fonctionnant en télé-traitement

Jamin, R.

Award date:
1975

Awarding institution:
Universite de Namur

FACULTES UNIVERSITAIRES NOTRE-DAME DE LA PAIX

Institut d'Informatique

Année académique 1974-1975

ETUDE DES TECHNIQUES DE RECOUVREMENT DE FICHIERS DANS UN SYSTEME D'ORDINATEURS FONCTIONNANT EN TELE-TRAITEMENT

Roland JAMIN

Mémoire présenté en vue de l'obtention du grade de Licencié et Maître en Informatique.

à mes Parents.

Je remercie les personnes m'ayant aidé à l'élaboration de ce mémoire et particulièrement Mr André DELAUNNOIS dont la collaboration me fut précieuse.

TABLE DES MATIERES

++++++++++++++++++++

PARTIE1. CONTEXTE GENERAL DU RECOUVREMENT DES FICHIERS

11 Introduction.....1.1
11.1 Disponibilité et fiabilité des systèmes.....1.1
11.11 Configurations de base.....1.3
11.12 Autres paramètres.....1.5
11.121 Fréquence des pannes.....1.5
11.122 Durée des pannes.....1.5
11.123 Intégrité de la data base.....1.6

12 Pannes dans un système on-line.....1.8
12.1 Erreurs du processeur.....1.8
12.2 Erreurs de parité mémoire.....1.9
12.3 Pannes dans le réseau de communication.....1.12
12.4 Pannes dans la périphérie.....1.14
12.5 Erreurs des opérateurs.....1.15
12.6 Erreurs de programmation.....1.16
12.7 Pannes de courant.....1.16
12.8 Pannes causées par l'environnement.....1.16
12.9 Saturation.....1.17
12.10 Pannes inexplicables:"acts of god".....1.17

13 Approches de recouvrement.....1.18
13.1 Halte complète du système.....1.18
13.2 Arrêt du terminal, de la tâche sources d'erreurs.....1.19
13.3 Fonctionnement en mode dégradé.....1.20
13.4 Aiguillage vers un sous-système stand-by....1.21
13.5 Aiguillage vers un système stand-by.....1.21

PARTIE2. ETUDE THEORIQUE DES TECHNIQUES DE RECOUVREMENT DES FICHIERS

21 Introduction.....2.1
22 Fall-back & switchover.....2.3
22.1 Fall-back.....2.3
22.11 File fall-back.....2.3
22.2 Switchover.....2.7

23 Recouvrement en batch.....2.11
23.1 Checkpoints.....2.11
23.2 Granfather, father, son tapes.....2.11

24 Recouvrement en on-line.....2.14
24.1 Dual recording of data.....2.14
24.2 Disks dumps.....2.15
24.3 Journaling.....2.18
24.4 Audit trail.....2.25
24.41 Texte de la transaction.....2.25
24.42 Texte de la transaction et copie des articles avant mise à jour.....2.28
24.43 Texte de la transaction et copie des articles après mise à jour.....2.30
24.44 Texte de la transaction + copie des articles avant màj + copie après màj.....2.31

PARTIE3. DEVELOPPEMENT D'UNE METHODE DE RECOUVREMENT DES FICHIERS

31 But de l'étude.....3.1

32 Conception générale du système.....3.2

33 Développement de la méthode.....3.3
33.1 Système global.....3.4

33.2 Programmation.....3.6
33.21 Enchaînement des programmes.....3.6
33.22 Programme principal.....3.6
33.23 Programmes applications.....3.13
33.24 Programme de sauvetage.....3.13
33.25 Programme de checkpoint.....3.19
33.26 Restauration.....3.21

34 Conclusion.....3.26

Lecture des listings.....3.27

DEFINITIONS

++++++++++++++++++++++

STAND-BY: se dit d'un second élément d'un système ou d'un second système entier pour un redémarrage rapide de l'exploitation.

SWITCHOVER: aiguillage vers un second élément d'un système ou vers un second système entier.

FALL-BACK: fonctionnement dégradé d'un système afin de ne pas arrêter complètement l'exploitation.

FRONT-END: ordinateur utilisé comme interface entre les lignes de communication et l'unité de traitement.

ON-LINE: reception des inputs directement de l'endroit où ils sont créés et envoi des résultats où ils sont nécessaires.
Nous associons dans cet ouvrage on-line et télétraitement.

TEMPS DE REPONSE: temps nécessaire à l'unité de traitement pour réagir à une sollicitation et en fournir le résultat.

TEMPS REEL: un ordinateur fonctionne en temps réel s'il reçoit des données, les traites et expédie les résultats en un temps défini (temps de réponse)
Le temps de réponse varie selon les exploitations:
- contrôle de processus: milliseconde
- opération administrative: 1 heure ou parfois plus.

PARTIE 1

CONTEXTE GENERAL DU RECOUVREMENT DES FICHIERS

11 INTRODUCTION

Lors de l'élaboration d'un système de télé-traitement, il faut toujours avoir à l'esprit les concepts de capacité, disponibilité et fiabilité des systèmes.

-capacité: mesure de l'aptitude d'un système à traiter une application déterminée. Elle est parfois considérée comme synonyme de performance.

- disponibilité: fraction du temps pendant laquelle le système peut être employé à des fins productives.

En d'autres termes, la capacité mesure la possibilité du système à traiter un problème défini alors que la disponibilité exprime la probabilité de service en fonction de la nature du système.

11.1 DISPONIBILITE ET FIABILITE DES SYSTEMES.

Un système doit être en état de fonctionner assez longtemps et au moment opportun de manière à ce que les résultats soient obtenus en temps utile.

Cette notion de temps utile est exprimée plus précisément par les utilisateurs selon:

- " Je ne peut supporter plus de M arrêts du système par mois, retardant la sortie des résultats plus de N heures ou minutes."
- " Tout arrêt du système de plus de 24 heures affecte le fonctionnement de mon entreprise de manière insupportable."
- " Chaque jour, il doit exister une période de M heures pendant laquelle le système a une probabilité de fonctionnement correct supérieure à 99,5 %."

Les contraintes relatives à la fréquence des arrêts ou au temps de dépannage font partie de la notion de fiabilité et celles relatives à la quantité

de temps disponible ou productif correspondent à la notion de disponibilité. Fiabilité et disponibilité sont liées d'une manière biunivoque et sont les deux facettes de la préoccupation fondamentale des utilisateurs.

Ceci nous permet de définir plus rigoureusement la fiabilité:

-"C'est la probabilité avec laquelle le système donnera des performances satisfaisantes pour une période de temps donnée et dans des conditions d'utilisation déterminées." (1)

Un utilisateur s'attend à ce que le système soit en état de fonctionner et capable d'accomplir certaines tâches pendant un temps donné. Toute interruption inattendue aura un impact sur la capacité de traitement du système, mais certaines seront insupportables, d'autres non selon les exigences de l'application.

En terme de fiabilité, système et application sont indissociables: un utilisateur peut avoir pour le même système des objectifs très différents pour deux applications.

Exemples:

- Commutation de messages et interrogation de fichiers par terminal.
  Pendant le temps réservé à cette application, la probabilité d'avoir un arrêt du système de plus de trois heures doit être inférieure à 5%.
- Applications administratives, par lots et en différé.
  Le temps de travail productif, sur le mois et durant l'horaire réservé à ces applications, doit être au moins égal à 200 heures.

---

(1) JP WINDAL: cours de télétraitement. Chap X p.36

Le système étant composé d'un certain nombre d'unités installées dans un environnement déterminé et ayant chacune un taux de pannes spécifique; ces unités exécutant, sous le contrôle d'opérateurs, des programmes en vue d'accomplir un ensemble de tâches, il faut dans le contexte de fiabilité et disponibilité prendre en considération:

- Les appareils avec leur fiabilité propre.
- L' environnement dans lequel ils travaillent (installations physiques telles que conditionnement d'air, proximité du personnel de dépannage, etc...).
- Le software (programmes de contrôles, programmes applications).
- Les opérateurs et pupitreurs.

Les calculs de fiabilité d'un système doivent tenir compte de tous ces éléments.

Le but de cette étude n'est pas d'en développer la démarche d'élaboration. Nous en citons toutefois quelques aspects car les critères utilisés et les résultats obtenus peuvent être déterminants dans le choix de la technique de recouvrement du système et des fichiers.

11.11 Configurations de base.

On en distingue trois types dont la combinaison constitue la configuration d'un système quelconque.

- Type "série": chaque unité doit fonctionner pour que le système marche.(figure 1.1 a)
- Type "Mdont N": au moins N unités parmi les M unités du système doivent fonctionner pour que celui-ci tourne.(figure 1.1 b)
- Type "parallèle": au moins une unité doit fonctionner pour que le système marche. C'est un cas particulier du précédent où N=1.(figure 1.1 c)

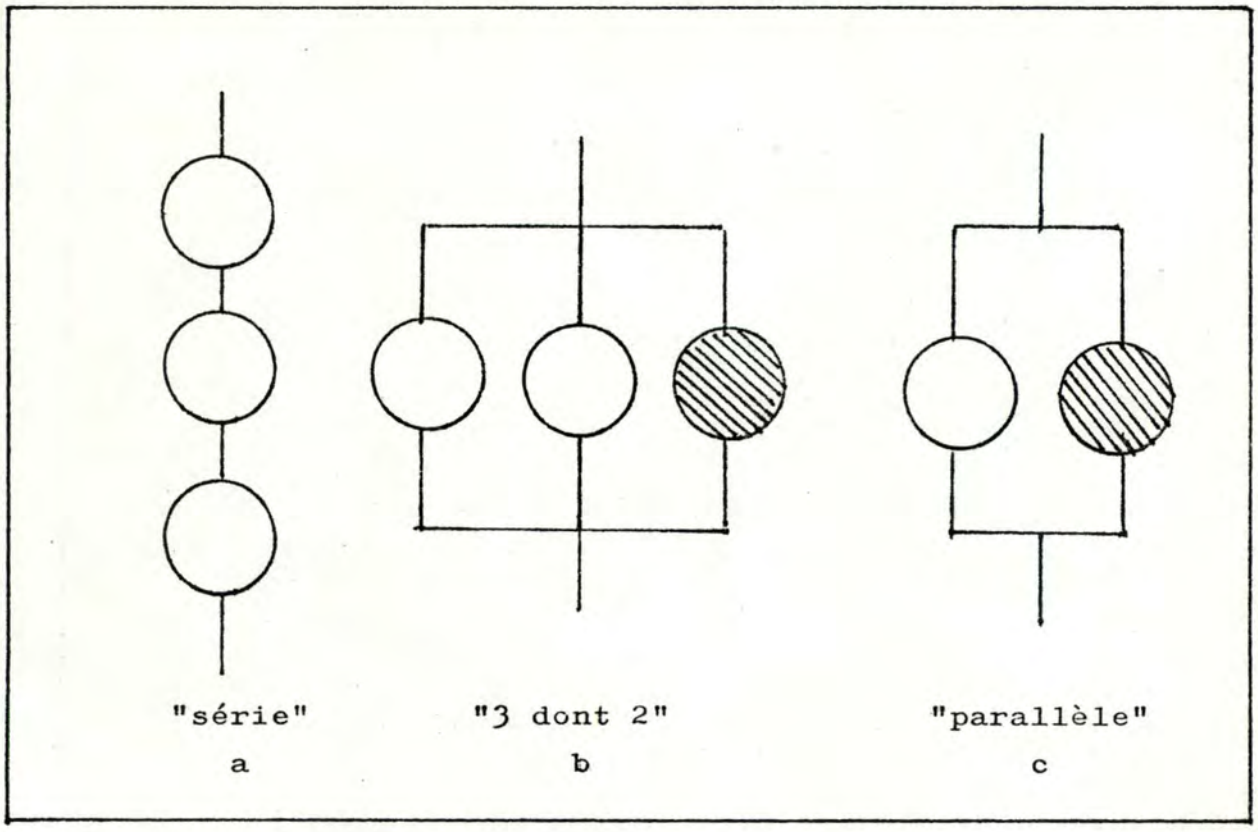


figure 1.1

Chaque type de configuration est l'objet d'une formule particulière. Nous donnerons pour mémoire:

- Type "série": $P = p1 \times p2 \times p3 \times \ldots \times pM$
  P = fiabilité du système
  pi = fiabilité de chaque unité

(1)

- Type "parallèle": $P = 1 - (1 - p)^M$
  Si toutes les unités ont la même probabilité de fonctionnement, $1 - (1 - p)^M$ est le seul cas où le système ne marche pas.(les M unités sont en panne)

---

(1) JP WINDAL: cours de télétraitement. Chap X p.39,40

11.12 Autres paramètres.

11.121 fréquence des pannes.

Certains systèmes on-line insistent sur la performance continue: contrôle de processus, trafic aérien, etc... N'importe quelle panne de n'importe quelle durée est catastrophique.

Exemple: Un processus de contrôle qui est hors service plus de quelques secondes peut entraîner une explosion ou une perte coûteuse.

La fiabilité d'un tel système est décrite par le temps moyen entre les pannes: M T B F (mean time between failures)

Il est généralement admis que l'apparition des pannes sur les unités d'ordinateurs suive la loi de Poisson, donc que le temps entre les panne tend vers une fonction de répartition exponentielle.

$$p(T) = e^{-\frac{T}{mtbf}} \qquad (1)$$

p(T) = probabilité pour que l'unité considérée fonctionne correctement pendant un temps au moins égal à T.
Si on connaît le MTBF de chaque unité, on peut calculer chaque p(T), puis le P(T) du système et de là son MTBF en passant par les logarithmes.

11.122 durée des pannes.

Prenons un système "business oriented". Etant en contact avec des utilisateurs humains, il peut endurer une ou deux pannes quotidiennes pour autant que leur durée soit courte. En effet, si la réparation est rapide, il ne se produit rien de fâcheux, mais dans le cas contraire,un client peut être perdu ou une transaction oubliée.

Le M T T R (mean time to repair) est le paramètre d'évaluation de la durée des pannes. Il inclut

tous les temps d'inactivité du système.

Exemple:

.5 min avant la détection de la panne et l'identification de son origine.

.5 min pour réparer

.10 min pour la restauration de la data base

.10 min pour relancer le système

La durée des pannes suit également une loi exponentielle:

$$d(t) = 1 - e^{-\frac{t}{mttr}}$$

d(t) = probabilité que la panne dure au plus un temps t
Connaissant le MTTR, on peut calculer le d(t) de chaque unité et le D(t) correspondant au système. On en déduira le MTTR global par les logarithmes.

NB fiabilité des programmes, environnement, personnel influencent aussi la fiabilité des systèmes.
(cfr JP WINDAL: cours de télétraitement. Chap X p.42,43)

11.123 intégrité de la data base.

Beaucoup d'organisations conçoivent actuellement des systèmes on-line avec des bases de données d'une taille telle qu'une centaine de bandes magnétiques sont nécessaires pour en garder une copie "back-up" et que leur vidage sur disque prend plusieurs jours.

D'autres organisations construisent des banques de données moins importantes, mais dont les opérations "jour à jour" constituent l'élément primordial. (Systèmes médicaux on-line,...)

Dans cet ensemble, les responsables des systèmes sont prêts à tolérer une panne par heure ou à rester hors service une grande partie de la journée pourvu que la data base reste intacte. Pour certains, par contre, le MTTR et l'intégrité de la data base sont insignifiants seul les intéresse le MTBF.(contrôle de processus p.e)

Le MTTR est le critère majeur dans les organisations pour lesquelles l'objectif principal est le niveau de service aux clients.

Généralement, on ne se borne pas à respecter un seul de ces trois éléments, mais deux (MTTR et intégrité de la base de données p.e) voire même les trois. Un système très bien conçu est celui où le MTBF et le MTTR sont excellents et l'intégrité de la data base respectée.

Ces différents concepts entraînent donc des procédures de recouvrement différentes.

Exemple:

MTBF: on se prémunit d'un hardware redondant. On suppose que la majeure partie des pannes sont du type hardware et que le software est capable d'effectuer un aiguillage immédiat vers le composant "back-up".

MTTR
INTEGRITE: développement d'un software facilitant un redémarrage rapide et une restauration aisée de la data base.
L'entraînement du personnel est très important car le meilleur software ne peut pas toujours prévenir l'opérateur du chargement d'une mauvaise version du système ou de la détérioration des procédures de recouvrement.

12 PANNES DANS UN SYSTEME ON-LINE

Le recouvrement se situe au niveau de tous les composants d'un système. Selon que telle unité est défectueuse, que la panne est de tel type, il y a lieu d'appliquer chaque fois la procédure de recouvrement adéquate. Nous exposons ici les différents types de pannes des systèmes on-line et les approches de recouvrement particulières afin de cerner le problème du recouvrement des fichiers, objet de la partie suivante.

Remarque: différence on-line/batch où on redémarre au début du programme ou à un point de contrôle.

Les causes des pannes sont identiques, mais la différence réside en la capacité à redémarrer le système.

En batch, on monte des rubans magnétiques et on insère des paquets de cartes.

En on-line, il faut passer par une procédure plus ou moins complexe pour s'assurer de l'intégrité de la data base, relancer le bon programme application et avertir les utilisateurs. Le software joue un rôle capital. Il doit détecter les types d'erreurs et prendre les mesures nécessaires pour les corriger, continuer dans un système dégradé ou le stopper.

Il est donc important de détecter le type d'erreur pour appliquer la procédure de recouvrement adéquate.

12.1 ERREURS DU PROCESSEUR.

Il peut arriver que le processeur n'exécute pas correctement les instructions. Un bit perdu peut fausser les références àdes zones particulières, l'exécution de certains branchements, etc...

Les erreurs du processeur sont généralement catastrophiques et on ne peut compter sur le software pour passer au processeur de réserve, s'il existe, communiquer avec l'opérateur ou exécuter une quelconque activité intelligente.

On y remédie par l'exécution d'un programme de diagnostiques contrôlant constamment le fonctionnement du processeur.

Cette solution - pleinement justifiée dans certains cas- présente cependant des inconvénients:

- temps CPU
- place mémoire

} dégradant les performances des systèmes.

12.2 ERREURS DE PARITE MEMOIRE.

Un bit de parité est associé à chaque mot ou à chaque octet en mémoire, rendant possible la détection de la plupart des erreurs de parité. Ce bit est adjoint automatiquement par le hardware à chaque stockage d'un mot ou d'un octet en mémoire et contrôlé lors de chaque lecture. La détection d'une erreur par le hardware déclenche une interruption et l'OS engendre une action particulière.

Ce type d'erreur est maintenant très rare (raffinement de la sécurité), mais catastrophique quand il survient.

Le type de recouvrement à appliquer dépend de l'organisation de la mémoire.

Exemple:

La mémoire est découpée en modules de taille fixe, les adresses sont croissantes à l'intérieur des modules, la succession de ceux-ci se fait dans l'ordre croissant de leur numéro.
(figure 1.2)

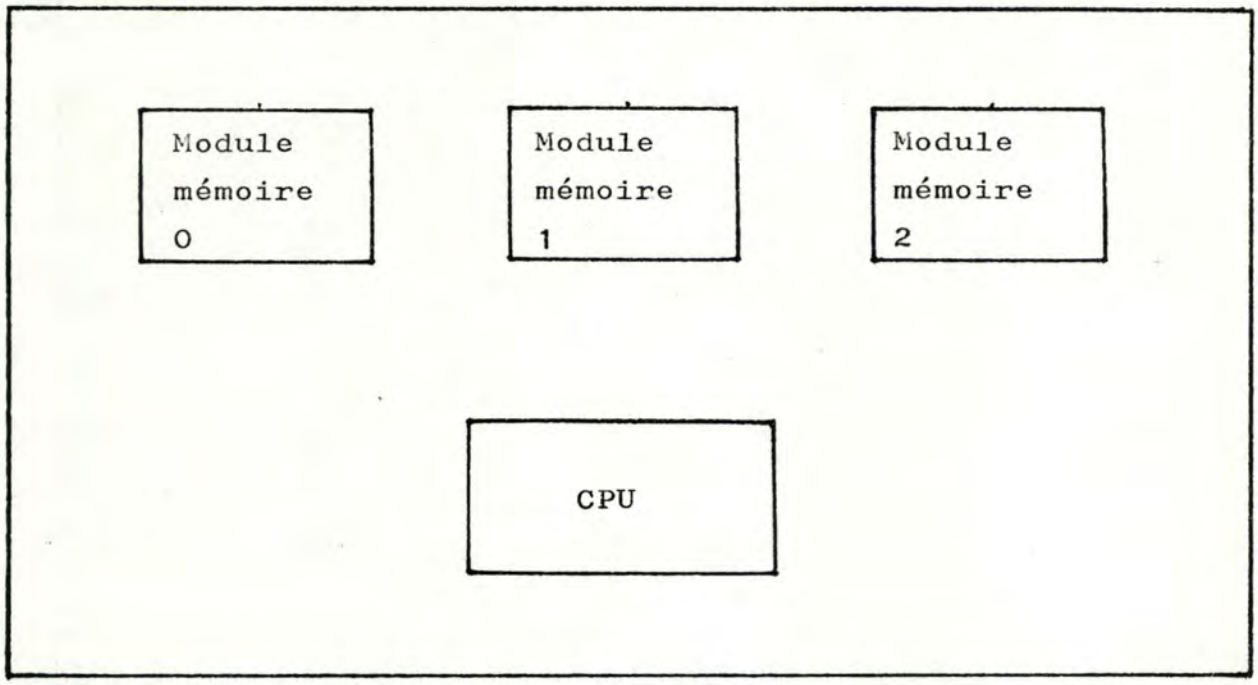


figure 1.2

Recouvrement.

1) Arrêter le système et reconfigurer manuellement la mémoire en changeant le numéro des modules (figure 1.3). Ceci implique une modification de la table des pages et des "core maps" afin d'éviter l'allocation du module fautif à un programme application. Le système est ensuite redémarré.

Pour exécuter ce type de recouvrement, le système doit être apte à tourner avec une mémoire partielle et doit pouvoir déterminer la taille mémoire disponible pour empêcher un adressage erroné.

figure 1.3:

L'OS constate une erreur de parité mémoire dans le module 1. Les numéros des modules suivant sont modifiés et les entrées dans la table des pages correspondant au module numéro 1(blocks du module n°1) sont marquées indisponibles.

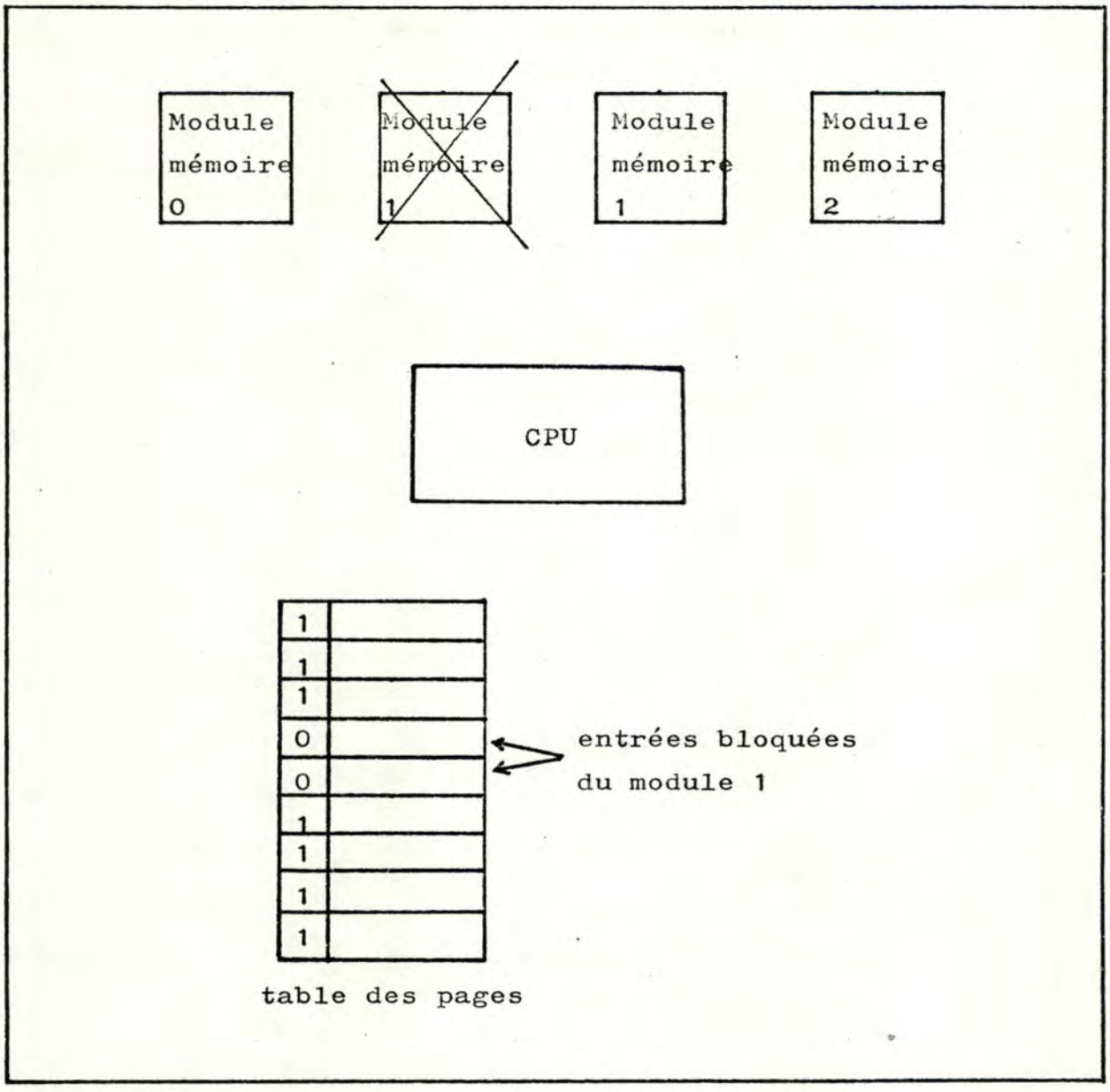


figure 1.3

2) Ne pas arrêter le système et exécuter la séquence suivante:
   - stopper l'exécution du ou des programmes applications dans le mauvais module.
   - appliquer un recouvrement des fichiers(voir 2ème partie)
   - si nécessaire, demander à l'utilisateur dont le programme s'exécutait dans le module fautif de retransmettre son dernier input. Si un ordinateur "front-end" est présent dans le système, on lui demande de retransmettre la dernière transaction.

- le ou les programmes applications sont rechargés dans un autre module et relancés.

NB Dès la détection de l'erreur, les utilisateurs doivent être prévenus de cesser de transmettre pour ne plus avoir d'input concernant le ou les programmes application en question avant leur relancement. Les opérateurs doivent aussi être informés des travaux pouvant continuer.

L'arrêt total ne pourra cependant pas être éviter dans le cas où l'erreur est constatée dans le module contenant le CAW, le PSW, etc...(module 0)

12.3 PANNES DANS LE RESEAU DE COMMUNICATION.

La figure 1.4 indique les endroits de survenance des pannes dans le réseau de télécommunication.

Les pannes de terminal, de ligne de communication à basse vitesse ou de leurs modems n'entraînent généralement pas une panne du système, à la différence de celles affectant un ordinateur "font-end", le multiplexeur, une ligne à haute vitesse ou leurs modems.

L'ingénieur système doit déterminer la mesure dans laquelle la panne d'un des composants du réseau de télécommunication perturbe le système tout entier. La panne d'un terminal peut être plus dangereuse que prévu s'il est en train d'exécuter une application très importante. En cas d'indisponibilité d'un terminal back-up, le système entier doit alors s'arrêter.

Exemple:

Une entreprise ayant un grand centre de stockage et un terminal unique affecté à la gestion du stock (entrées, sorties,...).

Si l'activité de l'entreprise est axée sur le stock, une panne du terminal enrayera tout le système.

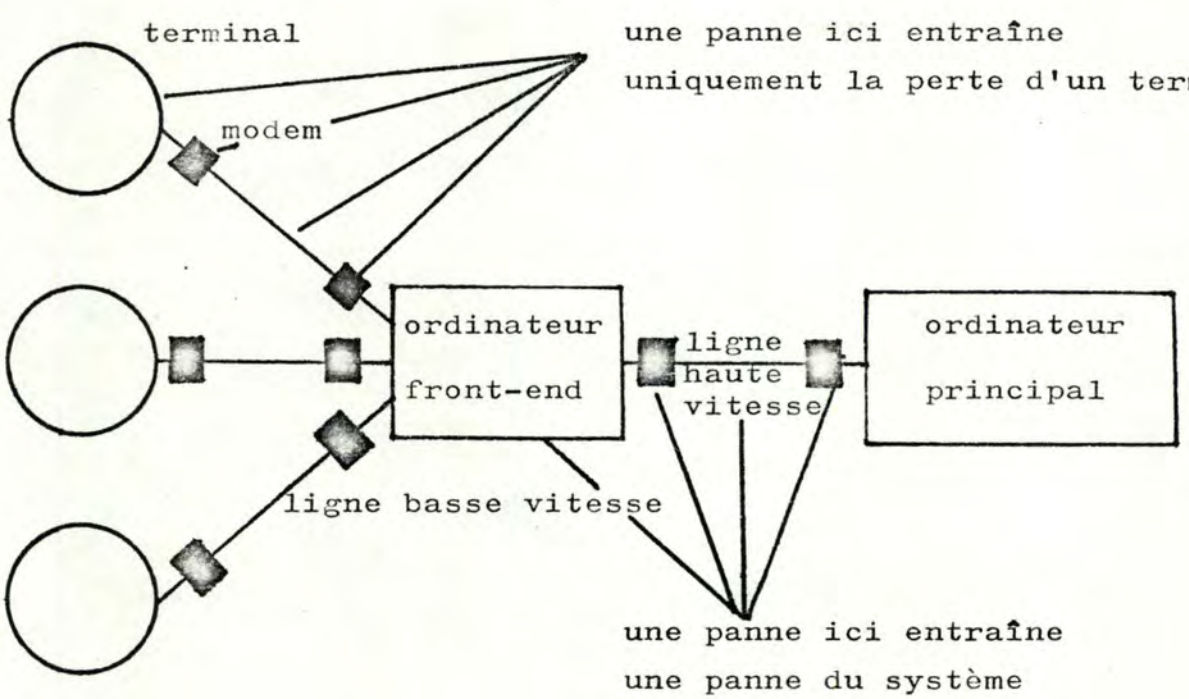


figure 1.4

Recouvrement.

- Terminal: on passe à un terminal back-up, sinon il faut attendre la réparation de la panne.(le terminal est mis hors circuit.)
- Ordinateur front-end: les causes de panne sont identiques à celles de l'ordinateur principal — panne processeur, erreur programme, erreur de parité — L'attitude à adopter est fonction de la cause de la panne.
- Ligne basse vitesse ou modem associé:
  la ligne est généralement du type commuté. Il suffit d'informer la centrale du dérangement de la ligne et d'en demander une autre.

- Ligne à moyenne et haute vitesse:

type loué, il faut alors s'assurer un autre chemin de communication en utilisant une ligne de secours par exemple.

Le facteur coût joue:

. au niveau de la ligne supplémentaire

. au niveau de la panne avec tout ce qu'elle implique

Comparer les deux coûts pour poser le choix du recouvrement.

12.4 PANNES DANS LA PERIPHERIE.

Ces pannes sont généralement transitoires et dues à la nature électromagnétique des appareils (saletés sur un ruban, oxyde sur les têtes de lecture/écriture, griffes sur les disques sont les types de problèmes les plus courants).

Elles n'engendrent pas un arrêt complet du système:

- elles sont transitoires. En recommençant une ou plusieurs fois l'opération d'IO, on parvient à les éliminer.
- les systèmes sont de moins en moins dépendants des appareils périphériques.En cas de panne d'un dérouleur de bandes ou d'une armoire à disques, le recouvrement s'effectue en montant la bande sur un autre dérouleur ou les disques dans une autre armoire.

Les situations les plus critiques sont donc les suivantes:

- article définitivement illisible (shift,garnissage de valeurs erronées...)
- head crash: écrasement des têtes de lecture/écriture sur la surface des disques.
- panne canal.

On applique pour les deux premiers cas une technique de recouvrement de fichiers vue dans la partie suivante.

Une panne canal doit toujours être accompagnée d'une indication d'erreur afin d'éviter tout dommage à la data base.

12.5 ERREURS DES OPERATEURS.

Certaines manipulations exécutées par les opérateurs au cours de l'exploitation sont accidentellement fautives.

On distingue quatre grandes classes d'opérations effectuées par les opérateurs et dans lesquelles une erreur peut entraîner une panne du système:

- démarrage du système
- exécution normale du système
- pendant et après la panne
- arrêt du système en fin de journée

Exemple: démarrage du système.

La procédure est longue et complexe:

. contrôler les conditions de fonctionnement correct de tous les éléments hardware

. chargement de la version adéquate de l'operating system

. chargement de la bonne data base, de la librairie correcte et des bons programmes applications

. utilisation du dialogue d'initialisation exact

. ouverture du dialogue utilisateurs-système

En fin de journée, l'opérateur doit s'assurer de la déconnection sans dommage des utilisateurs, effectuer un dump de la data base, etc...

Solutions:

1) entraînement intensif des opérateurs à faire face au plus grand nombre de situations possibles.
2) documentation précise et détaillée sur les procédures à suivre après l'erreur.
3) explications émanant du système sur les actions à entreprendre en cas d'erreur.

4) rendre le système le plus autonome possible en lui donnant des moyens de contrôle perfectionnés afin de limiter les interventions des opérateurs. Exemple du système capable de contrôler sa taille mémoire pour ne pas en dépasser la capacité.

12.6 ERREURS DE PROGRAMMATION.

Elles sont souvent difficiles à déterminer et causent des dommages à la data base.

Les programmes doivent être hautement testés pour en éviter le plus grand nombre au départ et si malgré cela une erreur est détectée, il faut appliquer une des techniques de recouvrement vues dans la deuxième partie.

12.7 PANNES DE COURANT.

Sont classées ici la panne totale et les variations de tension perturbant l'exploitation au niveau du système.

On empêche les variations grâce à des appareils fournissant une tension continue.

Un générateur autonome fournira le courant au système en cas de panne totale. Le système engendre une interruption lorsque la tension descend en-dessous d'un certain seuil. On sauve alors tout ce qui se trouve dans le système (fichiers, transactions,...). Une seconde interruption prévient du rétablissement de la tension à un niveau de fonctionnement correct.

12.8 PANNES CAUSEES PAR L'ENVIRONNEMENT.

Il s'agit du feu, de la climatisation défaillante, des inondations,etc...

12.9 SATURATION.

De graves ennuis peuvent être causés par un système manipulant un trop grand nombre de terminaux, d'interruptions, de transactions, ce qui entraîne une surcharge. Une telle situation rend l'ordinateur incontrôlable: dépassement des tables et des buffers, perte de transactions et de messages, dégradation de la data base.

Il faut donc prévoir la saturation et agir contre elle en demandant aux utilisateurs de ralentir leurs inputs, d'attendre un moment , en refusant de traiter une transaction pouvant entraîner un dépassement de la taille des fichiers, de la taille mémoire.

12.10 PANNES INEXPLICABLES: "ACTS OF GOD".

Un certain nombre de pannes demeurent inexplicables. Il arrive que le système s'arrête sans message d'erreur ni indication de la source software ou hardware de celle-ci.

Causes du mystère:

- documentation perdue ou erronée sans qu'on le sache
- variation de l'alimentation, indétectable humainement, avec défaut d'une interruption de ce type.

Il est donc très important de minimiser le nombre de ces pannes pour atteindre un niveau de sécutité élevé. Imaginons le cas où l'achat d'un système back-up est décidé pour pallier à des pannes mystérieuses dues en fait à des vices de manipulation ou à des perturbations électriques.

13 APPROCHES DE RECOUVREMENT

Nous parlerons ici des approches de recouvrement les plus courantes dans un système on-line. La méthode de recouvrement employée en cas d'erreur dépend du type de hardware disponible, du degré de sécurité exigé et des caractéristiques du système.

Les solutions ci-dessous ont été abordées dans le paragraphe 12, nous les détaillons.

13.1 HALTE COMPLETE DU SYSTEME.

Amener le système à un arrêt complet est à éviter le plus possible:

- confusion chez l'opérateur
  chez l'utilisateur
- des données ou des messages peuvent être irrémédiablement perdus
- des dommages physiques peuvent survenir dans la périphérie ou à d'autres appareils externes

L'arrêt complet doit être exécuté de telle sorte qu'il évite ces inconvénients en informant l'opérateur et les utilisateurs sur ce qui se passe, en arrêtant les appareils périphériques sans ennui et en ne détruisant pas la data base.

Parmi les pannes débouchant obligatoirement sur cette solution, notons:

- erreur du processeur (cfr 12.1), si pas de processeur back-up
- erreur de parité mémoire, selon son organisation
- panne de courant,...

L'avantage de cette solution est sa simplicité et son économie. On évite le coût et les inconvénients d'un hardware redondant, des routines de recouvrement complexes. Si une technique de recouvrement de fichiers est implémentée, l'arrêt complet est valable.

En cas de panne sérieuse, le système est hors service pour une période de temps prolongée (chute de l'alimentation d'au moins trente minutes, écrasement des têtes de lecture/écriture sur les disques). Le MTTR devient alors beaucoup plus important que le MTBF, il empêche le recours à cette solution et aboutit à la décision de se servir d'un système back-up.

Notons ici que le système back-up n'implique pas un accroissement de la sécurité mais une grande facilité à redémarrer l'exploitation.

13.2 ARRET DU TERMINAL, DE LA TACHE SOURCES D'ERREURS.

Une fois déterminée la cause de l'erreur(tâche ou terminal) une forme de recouvrement est de supprimer l'exécution de la tâche ou de déconnecter le terminal en lui envoyant toutefois un message d'information.

Cette approche est donc valable pour un grand nombre d'utilisations:

- Un input est irrecevable à cause d'erreurs de transmission. L'opérateur au terminal en est informé et le terminal est coupé.
- Un terminal a demandé l'exécution d'un programme application et le système découvre une erreur d'entrée/sortie non réparable. Le programme application est arrêté et le terminal en est averti.
- Pendant l'exécution d'une transaction dans un système dédicacé, le programme application traite une référence ou une instruction illégales et entre dans une boucle sans fin. L'OS arrête le programme application, envoye un message à l'utilisateur (ne pas retaper la transaction avant la correction de l'erreur), relance le programme arrêté et continue le traitement des transactions des autres utilisateurs.

La critique de cette solution est qu'elle contourne simplement la difficulté. Il est cependant préfé-

rable de l'adopter que de se trouver dans la situation où tous les utilisateurs sont affectés par une erreur causée par un seul d'entre eux. Il est de plus probable que le système "sautera" à chaque traitement de la transaction fautive, le seul remède étant donc la correction de l'erreur.

13.3 FONCTIONNEMENT EN MODE DEGRADE.

Consiste à permettre au système de stopper les parties affectées par la panne et donc de fonctionner avec les seuls autres éléments.

Exemples:

1) En cas de panne d'un perforateur de cartes, d'un dérouleur de bandes, on peut diriger l'output vers un autre endroit jusqu'à réparation. On les accumule sur un fichier bande ou disque pour être ensuite restitué.
2) Si un module de mémoire est l'objet d'une erreur de parité le système est capable de continuer avec les modules restants. Cela augmente le swapping et les overlays, donc le temps de réponse, mais le système continue néanmoins à fonctionner.
3) Processeur principal hors service: les transactions sont acceptées et mises en file d'attente par un ordinateur front-end s'il existe. Le traitement des transactions est retardé et le temps de réponse allongé, mais l'activité aux terminaux continue.
4) Si l'ordinateur front-end ou le réseau de communication est défaillant, les utilisateurs préparent leur inputs en off-line (cartes, bandes perforées,...). Ces inputs sont envoyés en masse une fois le réseau remis en activité ou amenés manuellement à l'ordinateur.
5) Une partie de la data base endommagée ou un programme application inexécutable entraînent une interdiction pour les transactions d'utiliser la partie de la data base ou le programme application en question. Les uti-

lisateurs en seront avertis mais les autres types de transactions continueront leur traitement.

La difficulté est de permettre à l'ordinateur de reconnaître q'un appareil en panne est remis en service. Il faut pour cela que l'opérateur ait un moyen d'avertir l'ordinayeur qu'une opération ou un composant précédemment interdits sont maintenant disponibles.

13.4 AIGUILLAGE VERS UN SOUS-SYSTEME "STAND-BY".

Un des objectifs principaux de l'exploitation est la performance continue du système. Il est nécessaide lui ajouter des éléments redondants tels que multiplexeur, terminaux, lignes de communications, ordinateur front-end ou n'importe quel autre élément spécialisé.

13.5 AIGUILLAGE VERS UN SYSTEME "STAND-BY".

Généralisation du point précédent. Un système entier supplémentaire est disponible — processeur,data base, unités périphériques, appareils de communication, source de courant,... —

Cette approche n'est envisageable que lorsque le MTBF est le critère le plus important car le coût de cette solution est exhorbitant. Elle trouvera son application plus facilement dans de petits systèmes (saisie de données, p.e) pouvant supporter la dépense d'un second mini-ordinateur et de son équipement.

Cette approche prend deux formes:

- système dual.

Les deux systèmes -primaire et dual- effectuent le même travail en parallèle pour que le dual puisse continuer en cas de panne du primaire.

- système duplex. (figure 1.5)

Un back-up existe pour chaque composant du système mais il n'est pas utilisé sans une panne du premier.
Un système duplex est plus économique. Quand les deux fonctionnent parfaitement, le second peut être utilisé à un travail différent du premier (batch, états de sortie,...).
Les processeurs des deux systèmes sont reliés par un canal qui leur permet de s'interrompre mutuellement.
Chaque système est donc constamment informé de l'état de fonctionnement de son vis à vis par l'envoi réciproque de messages. Ces messages sont initialisés par la machine primaire qui les envoye à intervalles réguliers (1/60 seconde p.e). La machine back-up les reçoit et envoye une réponse qui est en fait du même type que celui reçu, relatif au fonctionnement du 2e système.
Le dialogue continue jusqu'à l'arrivée d'un des événements suivant:

- Le CPU primaire arrête soudainement ses messages.
- Le CPU primaire détecte une panne et l'indique au second CPU.
- Le CPU back-up arrête soudainement ses messages.
- Le CPU back-up détecte une panne et l'indique au CPU primaire.

1er cas

Le CPU back-up réalise après deux intervalles de temps que le premier CPU ne fonctionne plus. Il doit arrêter tout travail en cours et reprendre celui exécuté par le premier système. Il faut pour cela que les messages soient suffisamment explicites sur la nature de la tansaction, le type de programme application traitéspar le premier système afin d'en informer le second. La périphérie est switchée sur le second CPU et l'exploitation continue après restauration des fichiers dans un état correct.

2e cas

Panne d'un composant du premier système (imprimante p.e). Le CPU primaire se switche sur l'imprimante back-up en avertissant le second CPU que son imprimante lui est ôtée. Celui-ci continue son travail en sachant qu'il ne peut plus rien écrire, envoye ses messages imprimante dans un fichier et les écrit une fois que l'imprimante back-up lui est restituée.
Il en va de même pour tout autre composant.
Notons que si la panne a pu détériorer la data base, il faut en effectuer le recouvrement.

3e cas

Après deux cycles, le CPU primaire se rend compte que le second est en panne. Il faut alors souhaiter que le premier CPU ne s'arrête pas, ce qui constituerait une impasse.

4e cas

Le CPU primaire sait qu'il ne pourra pas compter sur le composant back-up en panne au cas ou le sien deviendrait hors service.

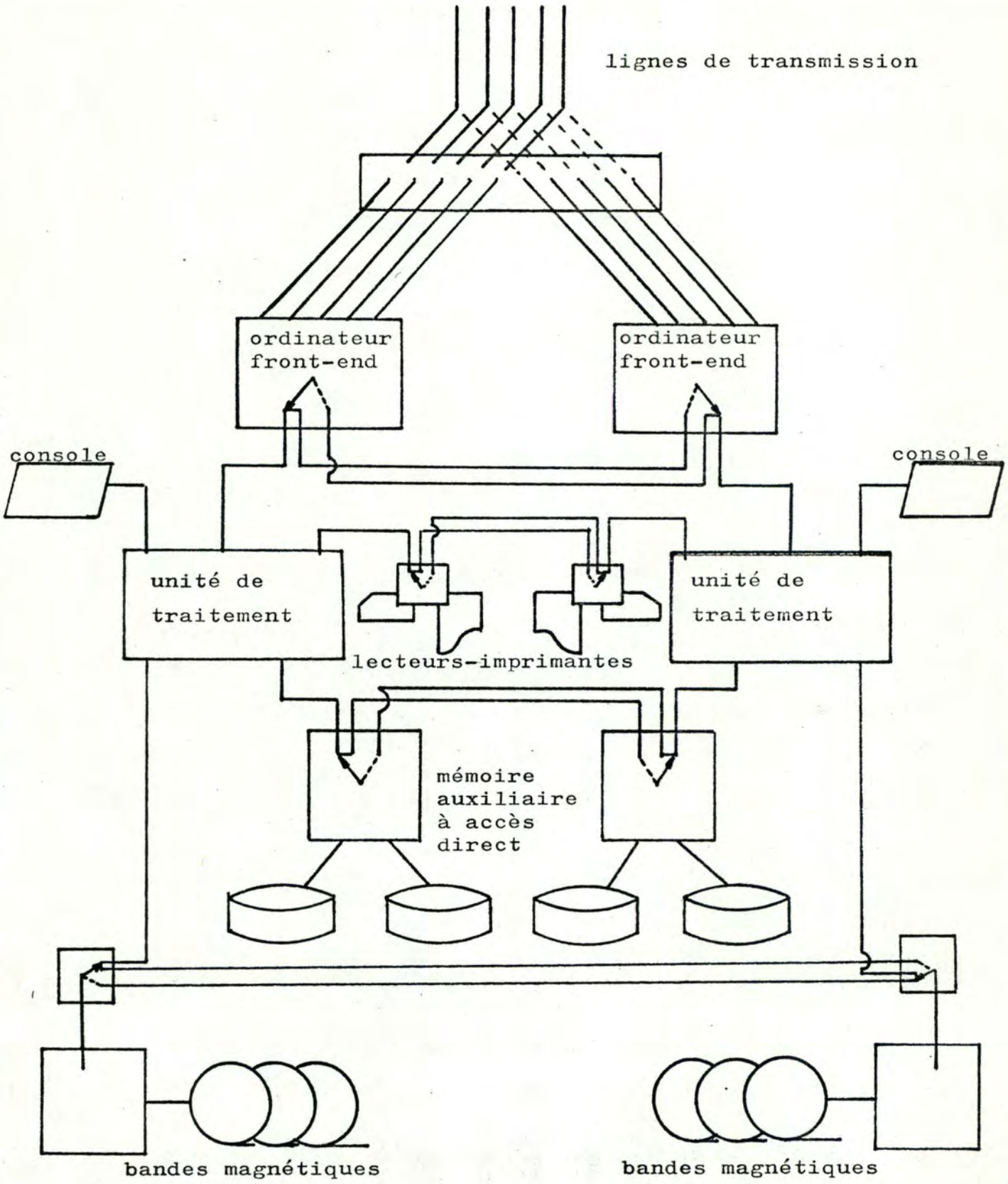

lignes de transmission
ordinateur front-end
ordinateur front-end
console
console
unité de traitement
unité de traitement
lecteurs-imprimantes
mémoire auxiliaire à accès direct
bandes magnétiques
bandes magnétiques

figure 1.5

PARTIE 2

ETUDE THEORIQUE DES TECHNIQUES DE RECOUVREMENT DES FICHIERS

21 INTRODUCTION

Nous considérerons les méthodes ci-après exposées en tenant compte des deux grandes classes de pannes et des deux types de destruction de fichier qu'elles engendrent.

- panne sérieuse: panne détruisant la data base (crash des têtes de lecture/écriture sur les disques, perte massive d'informations)
- panne triviale: qui ne provoque aucun pu peu de dommages à la data base.

- destruction logique: incidents survenus à la base de données suite à une panne triviale. Il faut alors reconstituer la data base grâce aux informations sauvées afin de pouvoir recommencer ou achever les transactions non terminées au moment de la panne et redémarrer ensuite l'exploitation.
- destruction physique: pertes de données suite à une panne sérieuse: cassure de disques après une chute, griffes, écrasement des têtes de lecture/écriture sur la surface magnétique.

NB. La perte massive d'informations d'origine software (cas où des erreurs de programmation ne sont pas détectées avant logtemps) n'entraînent pas une destruction physique au sens propre, mais bien une destruction logique telle que la procédure de recouvrement est presque identique à celle à adopter lors d'une destruction physique.
Nous l'appelons aussi destruction logique massive des fichiers.

La table 2.1 indique les causes possibles de destruction de fichiers, les moins probables figurant dans le bas.

causes de la destruction	extension de la destruction: partie d'un article	article entier	groupe d'articles	fichier entier
1. clé mauvaise	*			
2. erreur d'entrée venant d'un terminal	*			
3. erreur de transmission de données	*			
4. carte abîmée à la lecture		*		
5. erreur de programmation	*	*	*	*
6. accident durant le test d'un programme	*	*	*	*
7. chargement et mise à jour d'un mauvais volume			*	*
8. erreur d'un opérateur		*	*	*
9. bande ou disque égaré				*
10. bande ou disque défectueux	*	*	*	*
11. vol				*
12. détournement d'informations	*	*	*	
13. "manque de sérieux du personnel"	*			
14. feu, inondation				*
15. sabotage			*	*

table 2.1

22 FALL-BACK & SWITCHOVER

Nous exposons dans ce chapitre deux méthodes non axées fichiers mais s'y appliquant aisément.

22.1 FALL-BACK.

Consiste à assurer un fonctionnement en mode dégradé (cfr 13.3). Le système n'exécute que les parties vitales d'un job ou les jobs les plus urgents en n'utilisant que les unités fonctionnant correctement.

Exemple:

Deux lignes du réseau sont essentielles et deux disques contiennent les données primordiales. Le système n'accepte en cas de panne que les transactions passant par ces lignes et travaillant sur les deux disques.

L'idée d'une telle pratique est d'assurer un service plus ou moins continu plutôt qu'un arrêt total du système. La figure 2.1 donne quelques exemples.

Situations 3 et 4:

- l'ordinateur principal s'arrête: le LCC peut répon- aux messages essentiels à partir d'un lot limité d'informations maintenu dans ses propres disques.
- le LCC tombe en panne: deux lignes sont reliées à l'ordinateur principal et seules les transactions venant par ces lignes sont acceptées.

Situations 1,2,5:

- la data base est endommagée: on utilise les disques et tambours de réserve pour continuer l'exploitation On parle ici de file fall-back.

22.11 File fall-back.

Considérons un système dans lequel la data base occupe cinq modules physiques plus cinq autres pour les copies. Supposons que deux modules physiques au maxi-

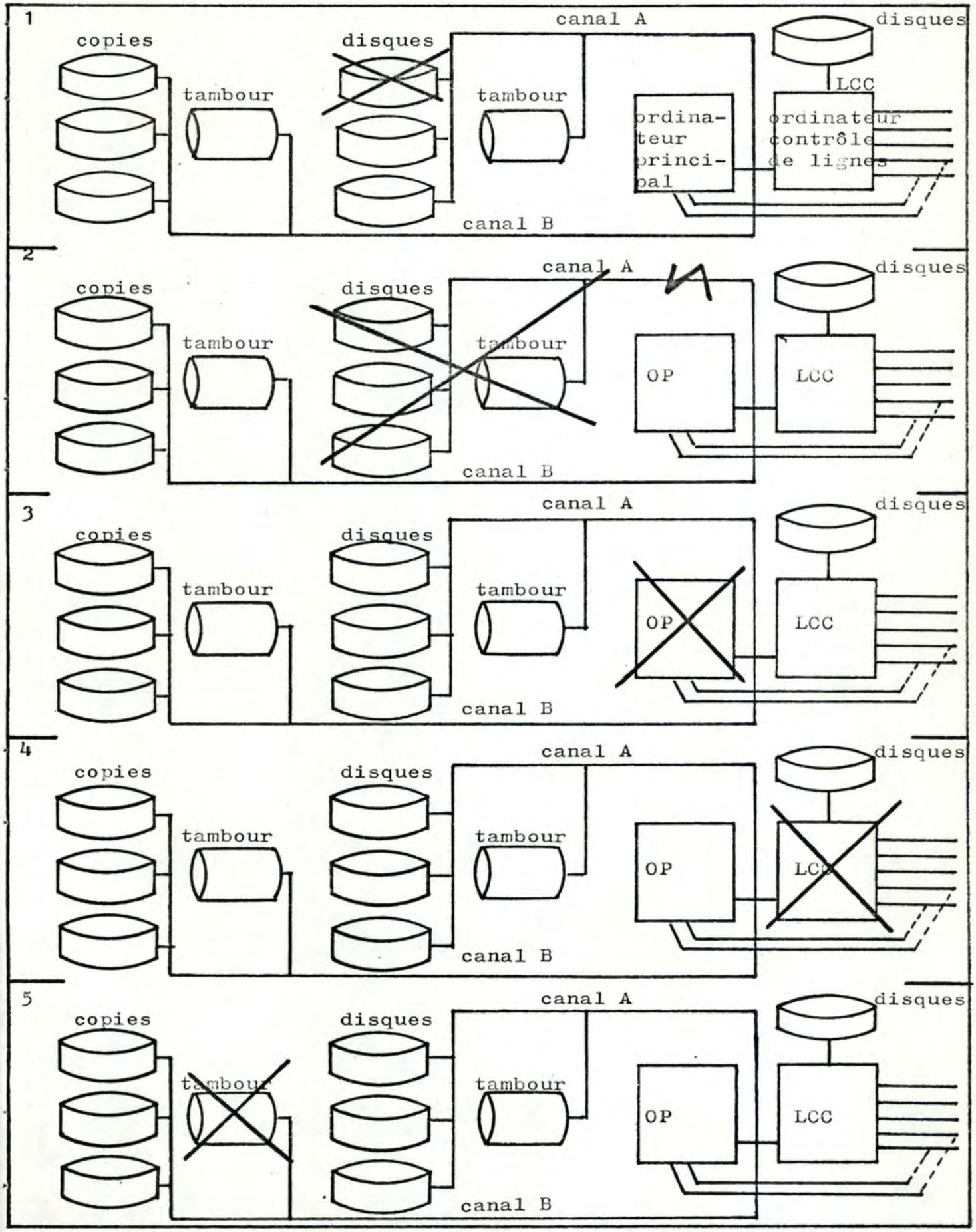

1
copies
tambour
disques
canal A
tambour
canal B
disques
LCC
ordinateur principal
ordinateur contrôle de lignes
2
copies
tambour
disques
canal A
tambour
canal B
disques
OP
LCC
3
copies
tambour
disques
canal A
tambour
canal B
disques
OP
LCC
4
copies
tambour
disques
canal A
tambour
canal B
disques
OP
LCC
5
copies
tambour
disques
canal A
tambour
canal B
disques
OP
LCC

figure 2.1

mum aient la possibilité de tomber en panne en même temps. Il faudra donc deux modules de réserve portant le nombre total à douze.

Les programmes application font référence à des modules logiques A, B, C, D, E (figure 2.2) en rapport avec la nature et le volume des données sans considération de leur stockage physique. Les programmes du superviseur maintiennent une table donnant les références des deux copies de chaque module logique (figure 2.3).

Exemple:

Le module n°4 est endommagé. Le module n°12 est alors copié sur le module de réserve n°5. Cette procédure est également suivie si un fichier est rendu indisponible par un entretien.(entretien du n°6, d'où copie du n°10 dans le n°9)

D'un point de vue utilisateur, l'impression de n'utiliser que cinq module reste intacte.

Problème:

L'écriture du second module sur celui de réserve s'effectue pendant le traitement des messages.

Les deux fichiers après copie ne seront pas dans le même état si des mises à jour surviennent dans la partie déja copiée.

Supposons pour la compréhension les articles portant les n°1, 2, ... , 10 et l'écriture du module 12 dans le module de réserve 5. La copie s'exécute dans la séquence des n° des articles et elle en est au n°8. Une mise àjour de l'article n°2 s'effectuera dans le module 12 mais pas dans le 5.

Un compte de progression de la copie est tenu dans la table pour résoudre ce problème. Il est consulté à chaque mise à jour durant la copie pour savoir s'il faut ou non la traiter dans le module récepteur (n°5).

Si le traitement des messages est interrompu pendant la duplication, les deux modules seront dans un état identique au redémarrage de l'exploitation.

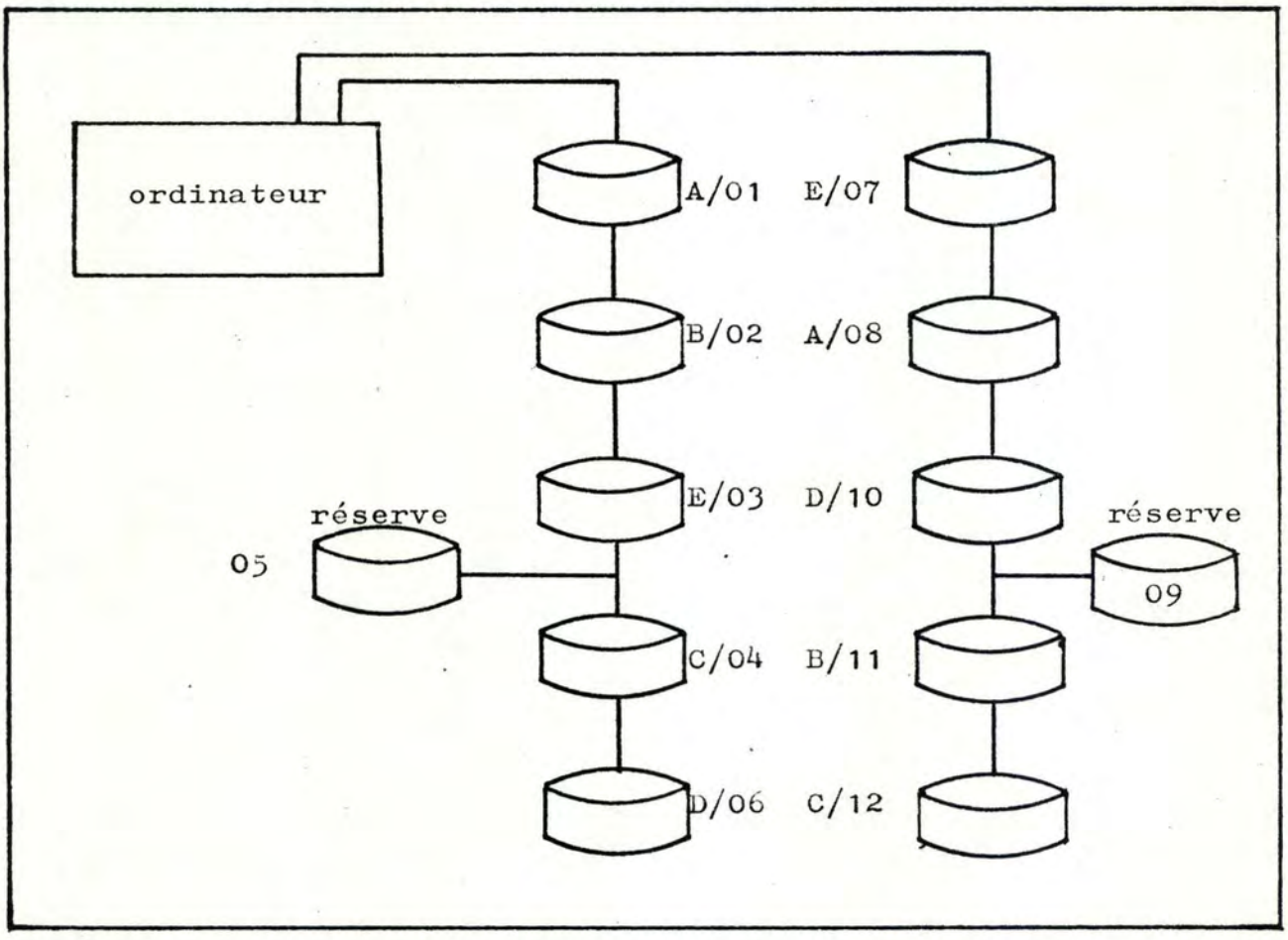


figure 2.2

fichiers logiques	copie1		copie2		compte de la progression de la duplication	
A	01	0	08	0		
B	02	0	11	0		
C	04	1	12	0	05	
D	06	1	10	0	09	
E	03	0	07	0		

statut: 0 : module disponible

1 : module indisponible

figure 2.3

Le fall-back présente l'inconvénient majeur d'être très cher si la base de données est importante. Un seul module physique pourrait être utilisé pour contenir uniquement les parties vitales des autres modules afin d'assurer un fonctionnement dégradé avec ces données essentielles, mais il faudrait alors prévoir une autre méthode de recouvrement pour restaurer les parties non vitales.

NB. Cette méthode s'applique aussi bien pour une destruction physique que logique de la base de données.
Une fois le module indisponible rendu opérationnel, il est réinséré dans l'ensemble comme module de réserve.

22.2 SWITCHOVER.

Cette technique implique un dédoublement des éléments vitaux d'un système (cfr 13.5 et 13.6). En cas de panne d'un de ces éléments l'application est "switchée" vers l'autre soit manuellement, soit automatiquement.

- DASD duplexé (direct and sequential device).
  Les données sont maintenues sur deux dasd et chaque mise à jour est effectuée sur les deux fichiers.
- l'autre composant ne doit pas rester inactif. L'utilisation de deux computers implique le fonctionnement du premier en on-line et du second en off-line (tests de programmes,...) pour des raisons économiques.

Les considérations dominantes en matière de fichiers sont de ne pas perdre de transactions, de s'assurer que les articles ne soient pas mis deux fois à jour par la même transaction et qu'il n'y ait pas d'oubli de mises à jour. Le message est garni pour cela d'un numéro de séquence unique. Le numéro de chaque message mettant à jour un article est ajouté à celui-ci.

Lors du switchover, le numéro de séquence de la dernière transaction est comparé avec ceux figurant dans les articles concernés pour identifiés ceux déja mis à jour (numéros identiques).

Le numéro de séquence est ajouté:

- par l'ordinateur de contrôle de ligne s'il existe (ordinateur front-end)
- par l'ordinateur central

La séquence est la suivante:

1) l'opérateur du terminal envoye le message.
2) réception du message par l'ordinateur qui ajoute le numéro de séquence.
3) l'ordinateur envoye un message au terminal, donnant le numéro de séquence en question.
4) l'ordinateur met à jour les fichiers et écrit le numéro dans les articles.
5) l'ordinateur envoye un nouveau message quand la mise à jour est satisfaite.

Une panne survient entre le premier message à l'opérateur et le second. Ce dernier message n'est pas reçu au terminal, mais le numéro de séquence est connu. L' opérateur envoye alors un message de contrôle avec ce numéro et l'ordinateur examine si les articles ont effectivement été mis à jour par comparaison des numéros de séquence. Ceci se déroule évidemment après le redémarrage.

Si la panne se situe avant le premier message à l'opérateur ou après le second, toutes les mises à jour ont été traitées.

Le traitement peut être automatique si un ordinateur de contrôle de ligne est utilisé (LCC). Le n° de séquence est établi par le LCC et ensuite stocké dans le computer principal dans le message reference block. Il n'est pas détruit avant la mise à jour et l'envoi du dernier message au terminal. (figure 2.4)

Dans ce cas, deux formes de switchover:

- si le computer principal est switché, le "stand-by" computer doit examiner la table des transactions dans le LCC et contrôler que chaque article a été ou non mis à jour dans les fichiers appropriés.

- si le LCC est switché, l'ordinateur principal doit à nouveau envoyer le message entier pour s'assurer que l'opérateur au terminal reçoive une réponse.

Une absence de réponse signifie que la transaction n'a pas mis à jour les fichiers, l'opérateur doit la retransmettre.

Problèmes:

1) Le LCC et l'ordinateur principal tombent en panne en même temps. En cas de conservation du contenu de leur mémoire, le recouvrement peut être exécuté, sinon il faut recourir à une autre méthode.
2) La technique du switchover coûte très cher. La dépense d'un deuxième système est supportable s'ils sont petits mais beaucoup moins dans le cas contraire, même en ne duplexant que le CPU.
3) Une panne dans le mécanisme d'écriture entre la mise à jour et le garnissage du numéro de séquence entraîne une grave perturbation de la data base.

Remarque:

Cette méthode est surtout préventive. Elle protège la data base d'un ennui survenant à un autre composant du système, mais elle n'est guère efficace si le dommage atteint les fichiers eux-même. Il faut alors recourir à une méthode de recouvrement de fichier proprement dite.

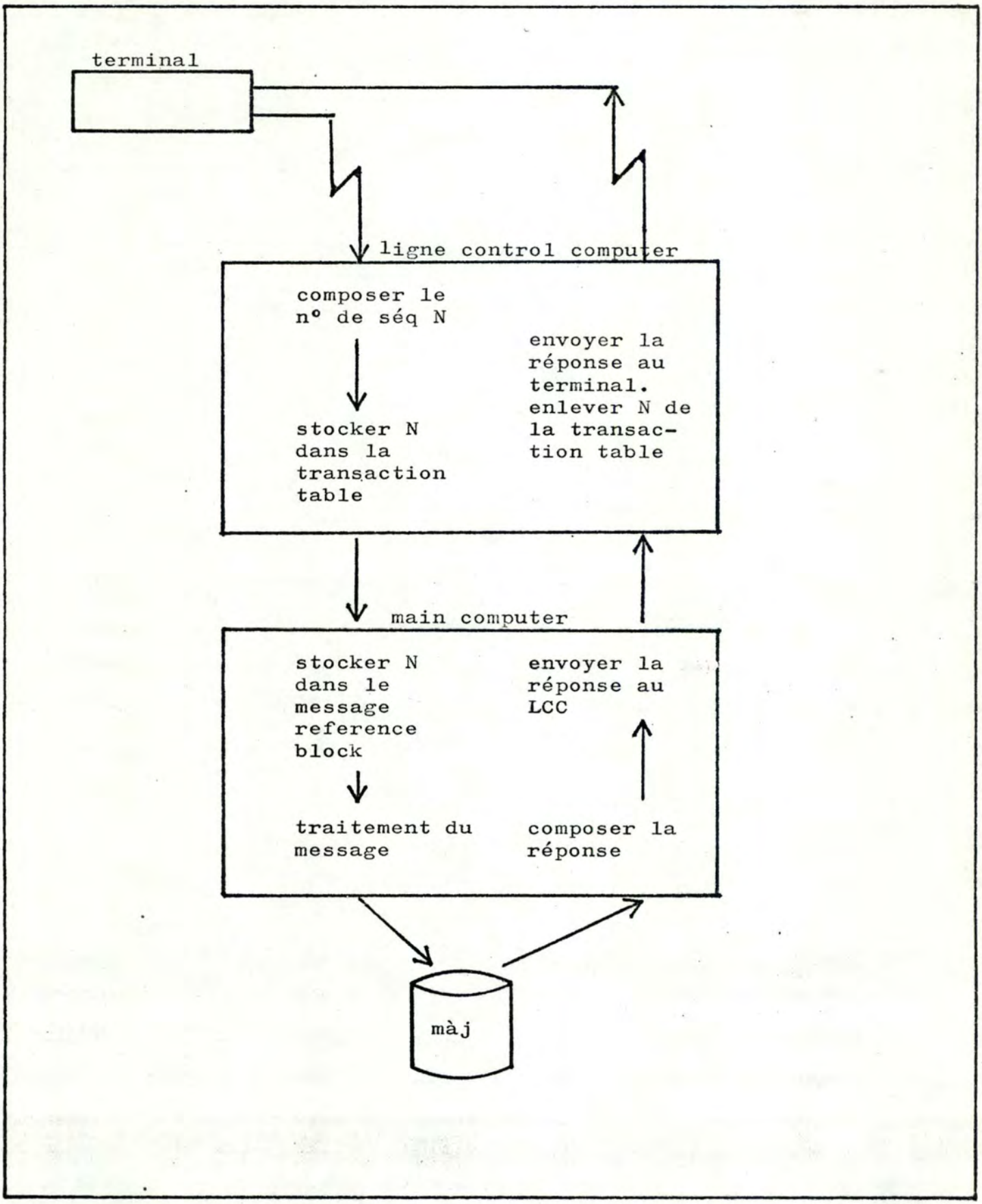

terminal
ligne control computer
composer le n° de séq N
stocker N dans la transaction table
envoyer la réponse au terminal.
enlever N de la transac-tion table
main computer
stocker N dans le message reference block
traitement du message
envoyer la réponse au LCC
composer la réponse
màj

figure 2.4

23 RECOUVREMENT EN BATCH

Les deux techniques principalement utilisées sont:

- les checkpoints
- grandfather,father, son tapes

23.1 CHECKPOINTS.

Cette technique n'est pas spécifique au batch et elle s'étend à un domaine plus large que les fichiers. Il s'agit de prendre à des instants déterminés des "photos du système (jobs, mémoire, fichiers). L'exploitation n'est pas relancée à son début, mais à partir de la dernière photo.

Nous étudions cette méthode plus en détails au paragraphe 22.5 et dans la 3e partie.

23.2 GRANDFATHER, FATHER, SON TAPES.

Chaque fonction de l'exploitation utilise en input un fichier appelé "père", et sort un fichier appelé "fils".

Voyons sur un exemple l'utilisation de cette technique. (figure 2.5)

Soit une entreprise recevant des commandes pour des produits qu'elle fabrique. Elle tient un fichier des commandes, un fichier des produits et un fichier des clients pour la facturation. L'exploitation effectue une fois par semaine la mise à jour du fichier des produits et la facturation (mise à jour fichier clients).

. Mise à jour fichier produits:

père: fichier produits de la semaine précédente
fils: fichier produits mis à jour. Il deviendra le père la semaine suivante.

. Facturation:

père: fichier clients semaine précédente
fils: fichier clients mis à jour

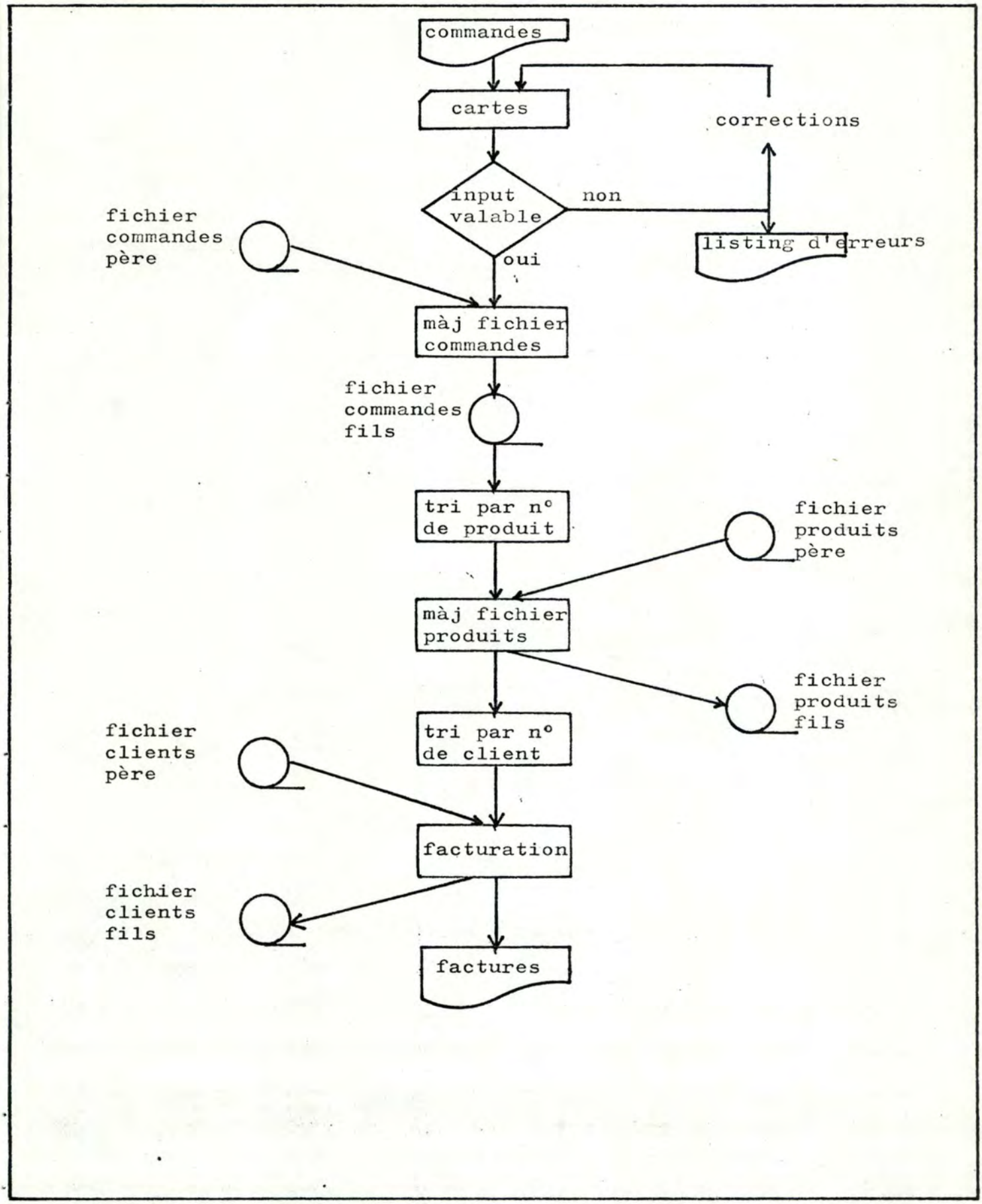

commandes
cartes
corrections
input valable
non
oui
fichier commandes père
listing d'erreurs
màj fichier commandes
fichier commandes fils
tri par n° de produit
fichier produits père
màj fichier produits
fichier produits fils
tri par n° de client
fichier clients père
facturation
fichier clients fils
factures

figure 2.5

On peut remonter plus loin dans l'arbre familial et utiliser un fichier "grand-père" pour se prémunir d'une perte du fichier père. Il faudra ranger ce fichier grand-père dans un endroit sûr à l'épreuve du feu, du vol etc...

Remarque:
Si l'exploitation utilise res fichiers à accès direct, les articles peuvent être "updated in place", c'est à dire que la mise à jour recouvre la zone précédente. Les fichiers grand-père, père et fils sont abandonné, il faut recourir à une autre technique de recouvrement.

Le recouvrement s'effectue en reprenant au début l'application pendant laquelle la panne est survenue et en utilisant le fichier père.
Destruction du fichier commande:
on prend le père, les cartes de commandes et on recommence la mise à jour
Destruction du fichier produit:
à partir du fichier produits père et du fichier commandes mis à jour, on recommence l'application.
Idem pour le fichier clients.
Ce recouvrement est valable pour les destructions physiques et logiques. En cas de destruction physique ou logique du fichier père, on le reconstitue en utilisant le fichier grand-père et les cartes relatives à l'application ayant traité le grand-père pour sortir le père.

24 RECOUVREMENT EN ON-LINE

Remarque préliminaire.

Parmi toutes les exigences et les opérations requises par les fichiers, l'intégrité d'une data base en est un aspect essentiel. En dépit de tous les efforts le système tombera occasionnellement en panne et il faudra y être préparé. De plus, une data base en télétraitement étant mise à jour dynamiquement, elle est plus vulnérable qu'en batch.

Il faut:

1) que la data base soit dans un état correct après la panne.
2) renseigner l'utilisateur sur son dernier input accepté ou lui donner des voies pour le retrouver.

Nous allons analyser les techniques les plus courantes de recouvrement de fichier dans un système on line en tenant compte de deux critères:

a) si le recouvrement ne garantit pas une data base correcte, il n'est pas utile.
b) s'il ne permet pas à l'utilisateur de déterminer l'état de son dernier input, son usage est limité.

24.1 DUAL RECORDING OF DATA.

L'idée est de garder la data base en deux exemplaires. Toute opération dans la data base (insertion, mise à jour, suppression) est effectuée simultanément dans chacun d'eux.

Cette méthode est très utile si les ennuis sont fréquemment associés aux disques. Une erreur de parité sur un disque ou un "head crash" détruira une data dase mais la seconde toujours intacte permettra de poursuivre les opérations. On reconstituera ensuite la data base fautive à partir de celle restée intacte.

Cette approche permet un recouvrement de la base de données, que la destruction soit physique ou logique.

Inconvénients.

1) Le "dual recording of data" protège la data base contre un nombre limité de pannes. Un grand pourcentage d'entre elles est dû à d'autres causes que celles touchant uniquement les disques, par exemple une chute de tension durant la mise à jour. Dans ce cas, les deux bases de données sont endommagées. Il en va de même lors d'une erreur dans un programme application de mise à jour.
2) Son coût est relativement élevé car on effectue chaque fois une double opération, il faut deux fois plus de devices périphériques.
3) Si la base de données est grande, il est physiquement impossible de la conserver en double exemplaire.

Une forme limitée de la "dual recording" approche peut être utile dans une organisation de fichiers par dictionnaires. Sa vulnérabilité est élevée car les dictionnaires sont les seuls indices de localisation des enregistrements. On ne garde un double que des dictionnaires car leur destruction engendre une grave perturbation de la base de données.

24.2 DISKS DUMPS.

Technique qui implique de copier la data base entière sur des rubans magnétiques ou des disks-packs, selon une base réglée. (période de temps).

Beaucoup de centre commerciaux ayant un système travaillant en on-line effectuent un dump à la fin de chaque journée.

En dépit de sa grande facilité, cette méthode présente des faiblesses:

- Une panne oblige la reprise du dernier dump et fait

perdre ainsi tout le travail accompli depuis lors. Il faut avoir sauvé les transactions et les réentrer au redémarrage de l'exploitation.
En prenant l'exemple d'un dump en fin de journée, une panne survenue tôt le matin du lendemain ne sera pas gênante, tandis qu'une autre en fin d'après-midi engendre la perte de tout le travail de la journée.

- Pour contourner cette difficulté: prendre un dump plusieurs fois par jour.
Cela s'avère impossible si la data base est grande.
Exemple: Supposons des devices rapides copiés au taux de 300.000 caractères/seconde.
En une heure, un millard de caractères sont copiés sur lesbandes.
Actuellement, les data bases commerciales contiennent jusqu'à 10 millard de caractères. Dix heures sont donc nécessaires pour effectuer un dump.
Un moyen de réduire le temps de copie est de stocker la base de données sur des unités indépendantes. Les dumps sont alors traités simultanément sur plusieurs bandes magnétiques.

La durée du dump est aussi inhérente à sa sophistication.
On distingue:

dump physique: dans lequel les tracks, les blocs ou les cylindres sont rangés séquentiellement.

dump logique: les articles logiques sont copiés dans un ordre séquentiel.

Même si le dump logique est plus lent, il présente des avantages. Il permet à la base de données d'être réarrangée et compactée. Le programme de dump peut contrôler les dictionnaires pour s'assurer de l'exactitude des enchaînements, des pointeurs et de la sauvegarde des fichiers. Il peut aussi contrôler la vraisemblance du contenu des zones de chaque article (zones numériques, alphanumériques, etc...).

Un mauvais enregistrement pouvant être ignorés plusieursjours, semaines et même quelques mois, il convient de conserver les dumps pendant un certain temps.

La conservation doit être ordonnée afin qu'un enregistrement erroné ne soit pas découvert après l'abandon du dump correspondant.

Démarche généralement suivie:

. cinq ou sept bandes quotidiennes
. quatre bandes hebdomadaires
. douze bandes mensuelles
. une bande annuelle

1) prendre un dump aprè chaque jour d'exploitation. Ils constitueront les cinq ou sept dumps quotidiens.
2) en fin de semaine, le dernier dump quotidien remplace le plus vieux dump hebdomadaire.
3) en fin de mois, le dernier dump hebdomadaire remplace le plus vieux dump mensuel.
4) en fin d'année, le plus récent dump mensuel devient un dump annuel gardé définitivement.

En d'autres termes:

- le dump du premier jour de la semaine en cours remplace le dump du premier jour de la semaine précédente.
- le premier dump hebdomadaire du mois en cours remplace le premier dump hebdomadaire du mois précédent.
- le premier dump mensuel de l'année en cours remplace le premier dump mensuel de l'année précédente.
- le dernier dump mensuel de l'année en cours devient un dump annuel conservé définitivement.

Variante:

Differential disk dump.

On conserve un dump pour les articles des fichiers ayant changé durant la journée ou un dump des articles qu'on sait sujet à de fréquentes mises à jour ou encore un dump des parties vitales des fichiers.

Cependant, reconstituer une data base complè-

te est plus difficile car il faut reprendre le dernier dump complet et le fusionner avec tous les dumps partiels effectués depuis lors.

Recouvrement:

Destruction logique: il faut mesurer l'extension du dommage afin de déterminer quel dump utiliser pour restaurer les fichiers.
La détérioration étant mineure, on repartira avec le dernier dump quotidien et le fichier des transactions
Dans certains cas, il faudra prendre un dump hebdomadaire - le dernier ou l'avant dernier - ou un dump mensuel vieux de quelques mois selon l'importance de la destruction.

NB. Les fichiers transactions étant important pour la restauration, il font partie intégrante de la data base et les mêmes précautions sont à prendre à leur sujet.

Destruction physique: la procédure est identique.

24.3 JOURNALING.

Le "journaling" consiste à tenir un fichier registre des transactions à traiter par le système et un fichier des articles de la data base avant et après mise à jour.

Utilité des journaux:

1) permettre de suivre le cheminement d'une transaction.
2) permettre un recouvrement des fichiers quand un utilisateur a mal mis à jour un ou plusieurs articles réutilisés ailleurs.
3) enquêter sur les causes de destruction ou d'endommagement d'un article.
4) recouvrir une destruction massive de fichiers.
5) aide à la correction d'un programme ayant perturbé la database.

6) aide au recouvrement après une panne du système.
7) permettre un recouvrement après la perte d'un journal.

Deux types de journaux:

- le journal des transactions
- le journal des actions dans les fichiers

Le premier contient les informations concernant les messages entrants et parfois les messages sortants. Le second contient les informations en rapport avec les mises à jour des fichiers.
Les deux types d'informations peuvent être gardés dans un seul journal, on parle alors " d'audit trail". Cette technique est expliquée plus en avant, elle est un peu moins efficace que la journalisation qui sépare les deux types de fichiers.

La journalisation permet:
a) d'accroître l'efficacité et les performances du système en raison du recouvrement rapide des fichiers.
b) le recouvrement d'un journal à partir de l'autre.

La synchronisation des événements est très importante. Quand un terminal met à jour un article, la séquence suivie est celle de la figure 2.6. On informe l'utilisateur qu'il doit réentrer sa transaction s'il ne reçoit pas de numéro de séquence pour celle-ci.

Le numéro de séquence envoyé par l'ordinateur au terminal permet la synchronisation entre les deux journaux.

Contenu des journaux:

1) Les tables 2.2 et 2.3 donnent une liste des informations pouvant figurer dans les journaux. Plus on veut faire face à un grand nombre d'éventualités, plus il faut enregistrer de renseignements et plus la complexité du programme de traitement des journaux est élevée.

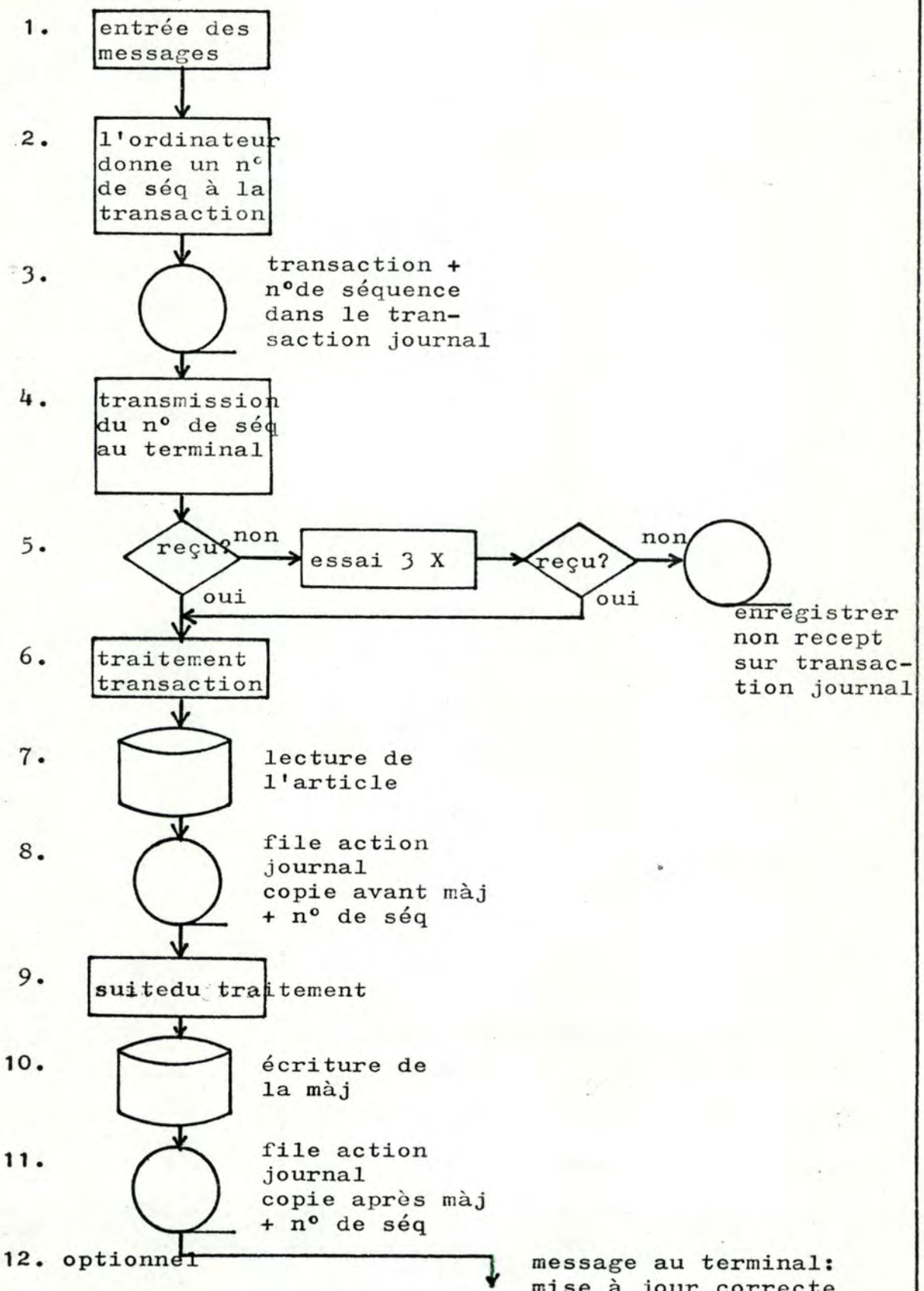

1.
entrée des messages
2.
l'ordinateur donne un n° de séq à la transaction
3.
transaction + n°de séquence dans le transaction journal
4.
transmission du n° de séq au terminal
5.
reçu?
non
essai 3 X
reçu?
non
oui
oui
enrégistrer non recept sur transaction journal
6.
traitement transaction
7.
lecture de l'article
8.
file action journal copie avant màj + n° de séq
9.
suitedu traitement
10.
écriture de la màj
11.
file action journal copie après màj + n° de séq
12. optionnel
message au terminal: mise à jour correcte

figure 2.6

2) Les journaux enregistrent historiquement toutes les valeurs de toutes les zones utiles de chaque article, de sorte que l'état d'un article à un moment précédent quelconque puisse être recréé.
3) Dans le journal des transactions figurent uniquement les messages de mise à jour de la base de données.
4) Les articles en entier ne sont pas toujours enregistrés dans le "file action journal", les parties essentielles sont souvent suffisantes.
   Exemple: fichier stock
   n° d'article, nom, ancienne balance, nouvelle balance
5) La synchronisation entre les journaux est résolue par un numéro de séquence ajouté à la transaction et enregistré dans chacun d'eux.

Recouvrement.

Destruction logique.

L e programme de recouvrement examine le journal des transactions et le journal des actions dans les fichiers selon la procédure suivante:

1) éliminer les transactions dont le numéro de séquence figure dans les copies "after" et celles dont le numéro de séquence n'a pas été reçu par le terminal (elles sont retransmises).
2) les transactions restantes sont traitées par le programme de recouvrement comme suit:
   a- transactions dont le numéro de séquence ne figure pas dans le file actions journal, copie avant mise à jour.
   La mise à jour est effectuée à partir du contenu de la base de données.
   b- transactions dont le numéro de séquence figure dans le file actions journal, copie avant.
   La panne a pu survenir entre la màj et la copie après. Recommencer la mise à jour à partir du contenu de la base de données provoquerait des erreurs.

Exemple: contenu d'un article: 200
màj: + 10
panne entre les points 10 et 11 figure 2.6
contenu base de données: 210 (màj effectuée)
reconstruction à partir de la data base:
210 + 10 = 220: erreur
reconstruction à partir de la copie avant:
200 + 10 = 210: correct

Il faut donc retraiter la mise à jour à partir de la copie avant, une fois la restauration terminée.

NB. Erreur constatées après un long laps de temps (erreurs de programmation p.e).
La sécurité exige de prendre périodiquement des dumps de la data base.

1) déterminer l'origine temporelle de l'erreur
2) prendre le dump précédent cette origine, l'historique du fichier des transactions et les retraiter toutes à partir de ce moment en suivant la séquence 6 à 11 (figure 2.6).

Destruction physique.

Utiliser le fichier des transactions, le dernier dump de la data base et recommencer le traitement avec les transactions subséquentes au dump, après fusion du file actions journal et du dump en utilisant les copies after pour éliminer les transactions entièrement traitées.

o = oui
p = possible
i = à certains intervalles

JOURNAL DES TRANSACTIONS

besoins \ actions	suivre l'histoire de la transaction.	recouvrement après mauvaise màj.	enquête sur les causes d'endommagement d'un article.	aide au recouvrement d'une destruction malvive.	aide à la correction de fichiers suite à des dégats venant des programmes.	correction de mauvaises informations envoyées aux utilisateurs du système.	contrôler les violations de procédures pour découvrir de possible infractions à la sécurité	aide au recoustrement après une panne système	contrôle de l'utilisation du système	recouvrement après perte d'un fil actions journal.
transaction "demande de renseignements " en input.						o			o	
transaction "mise à jour" en input.	o	o	o			o		o	o	o
type de transaction		p	p			o	o		p	
n° de transaction.	o	o	o			o		o		o
terminal origine.	o	o	o			o	o	o	o	o
opérateur "origine".	o	o	o			o	o		o	o
heure et date.	i	i	i			i	o	i	i	i
réponse à une transaction "demande d'enquête".						o			o	
réponse à une transaction "mise à jour".	o	o	o			o		o	o	o
indication d'une bonne réception de la réponse.	o	o	o			o		o		o
violations de la procédure d'input.							o			
enregistrement début et fin de reconstruction des fichiers.									o	
note de l'achèvement de la mise à jour.									o	

table 2.2

actions

o = oui
p = possible
i = à certains intervalles

FILE ACTIONS JOURNAL

besoins

besoins	Suivre l'histoire de la transaction.	recouvrement après mauvaise màj.	enquête sur causes d'endommagement d'un article.	aide au recouvrement d'une destruction massive.	aide à la correction de fichiers suite à des dégâts venant des programmes.	correction de mauvaises informations envoyées aux utilisateur du système.	contrôler les violations de procédures pour découvrir de possibles infractions à la sécurité.	aide au recouvrement après une panne système	contrôle de l'utilisation du système	recouvrement après perte d'un File actions journal.
n° de transaction.	o	o	o		o			o		
heure et date.		i	i	o	i					
adresse des items mis à jour.	o	o	o	o	o			o		
copie avant mise à jour.	o	o	o	o	o			o		
copie après mise à jour.	o	o	o	o	o			o		
note de l'achèvement de la mise à jour.	o	o	o	o	o			o		
liste des programmes utilisés pour l'update.			o		o					
contenu entier des articles créés.			o	o	o			o		
contenu entier des articles détruits.			o	o	o					
violations des procédures de mise à jour.							o			
contenu des items à corriger, avant correction.	o		o	o				o		
contenu des items à corriger, après correction.	o		o	o				o		
indication de début et fin de recouvrement.	o		o	o				o		

table 2.3

24.4 AUDIT TRAIL.

La philosophie de cette méthode est de garder un historique des inputs traités par le système et des changements intervenus dans la data base.

Dans cette voie, on peut facilement restaurer la base de données et retraiter un travail perdu.

Rappel:

Panne sérieuse: panne détruisant la data base (crash des têtes de lecture/écriture sur les disques, perte massive d'informations...)

Panne triviale: qui est sensée ne provoquer aucun ou peu de dommages à la data base.

Un audit trail consiste (cfr journaling) en un enregistrement, usuellement sur bande magnétique d'un ou plusieurs types d'informations suivant:

- texte des transactions entrées.
- copie des enregistrements avant mise à jour.
- copie des enregistrements après mise à jour.

A la différence du journaling, l'audit trail maintient ces renseignements sur un seul fichier.

24.41 Texte de la transaction.

En ne gardant uniquement que le texte de la transaction, l'audit trail prend la forme de la figure 2.7
D'autres renseignements viennent s'y ajouter:

- n° du terminal
- n° d'identification de l'utilisateur
- heure d'arrivée de la transaction,...

Nous sommes en possession de toutes les transactions acceptées par le système, mais nous ignorons la manière dont la data base a été transformée. En cas de panne, il faut donc supposer que le système était en cours de traitement de la ou des transactions les plus récentes, tout en ne sachant pas combien de traitements ont été

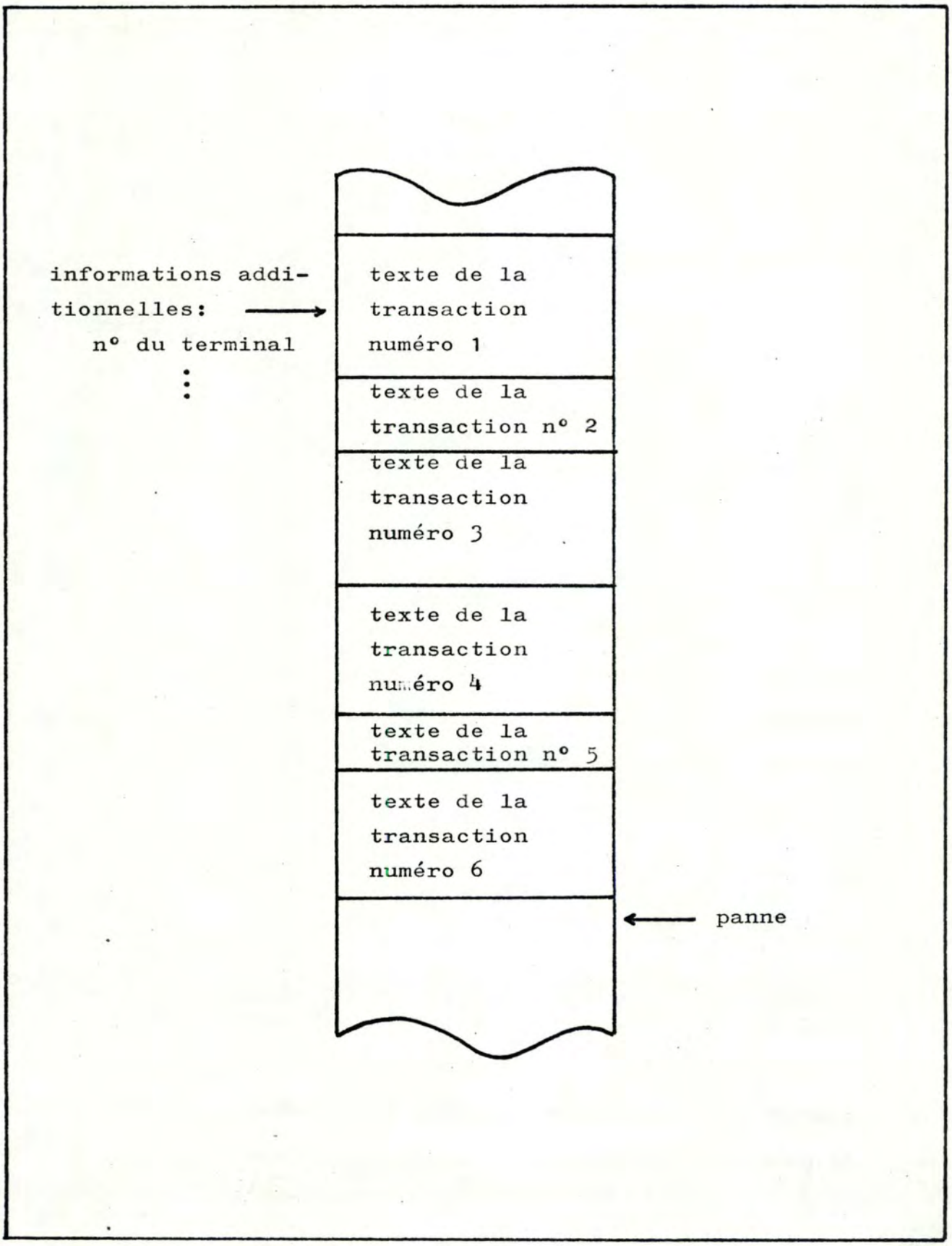

informations addi-
tionnelles:
n° du terminal
texte de la transaction numéro 1
texte de la transaction n° 2
texte de la transaction numéro 3
texte de la transaction numéro 4
texte de la transaction n° 5
texte de la transaction numéro 6
panne

figure 2.7

effectués ni quels enregistrements ont été modifiés.

Une solution est d'écrire un programme de recouvrement examinant d'une quelconque manière l'état courant des fichiers et déterminant ce que le programme application effectuait au moment de la panne.

Le programme de recouvrement pourrait:

- terminer le traitement commencé par le programme application
- ou bien le défaire.

La solution est fastidieuse. Le programme de recouvrement doit simuler le traitement normal du programme application et contrôler la base de données pour chaque transaction afin de voir si la mise à jour a déjà été faite ou non. Dans de nombreux cas, la simulation et le contrôle de la data base ne peuvent être exécutés par le programme de recouvrement.

Exemple:

Le programme application incrément ou décrémente les contenus d'une zone numérique dans les enregistrements de la data base. N'ayant aucune information sur les précédents contenus, le programme de recouvrement ne peut déterminer ni dans quel sens la mise à jour a été faite ni l'exécution de la mise à jour elle-même.

Recouvrement.

Destruction physique et logique.

Il faut recharger le dernier dump complet de la base de données et retraiter toutes les transactions enregistrées dans l'audit trail. Ne dépendant pas de l'état courant de la data base, la méthode convient en cas de destruction physique et logique, mais elle est inapte à un recouvrement valable si une des trois conditions ci-dessous est vraie:

1) la base de données est très grande, de sorte que le chargement du dernier dump complet prend un temps considérable.

2) un grand nombre de transactions ont été accumulées sur la bande audit trail et/ou les transactions réclament un temps CPU important.
3) des pannes surviennent souvent et les logiques de rechargement de la data base et de retraitement des transactions sont assez pauvres.

Solution auxiliaire:

Réserver dans chaque article de la data base une zone contenant le numéro de la dernière transaction ayant mis à jour l'article.

Cette information supplémentaire permet de retraiter les dernières transactions entrées dans le système avant la panne en utilisant les programmes applications normaux.

- informer les subroutines de manipulations de données et les programmes applications que le système travaille en mode recouvrement.
- utilisation de la zone contenant le numéro de transaction pour prévenir, par comparaison, les mises à jour redondantes lors du retraitement des transactions.

La procédure est fort longue et peut engendrer de nouvelles erreurs en cas de destruction du numéro.

24.42 Texte de la transaction et copie des articles avant màj.

Cette méthode implique l'enregistrement du texte de la transaction et de la copie avant mise à jour des articles. On indique dans la copie le numéro de transaction pour réaliser la correspondance entre cette transaction et toutes ses mises à jour. On ajoute aussi dans l'audit trail un enregistrement de fin de transaction. (figure 2.8)

Recouvrement.

·Destruction logique.

Le programme de recouvrement scanne la bande

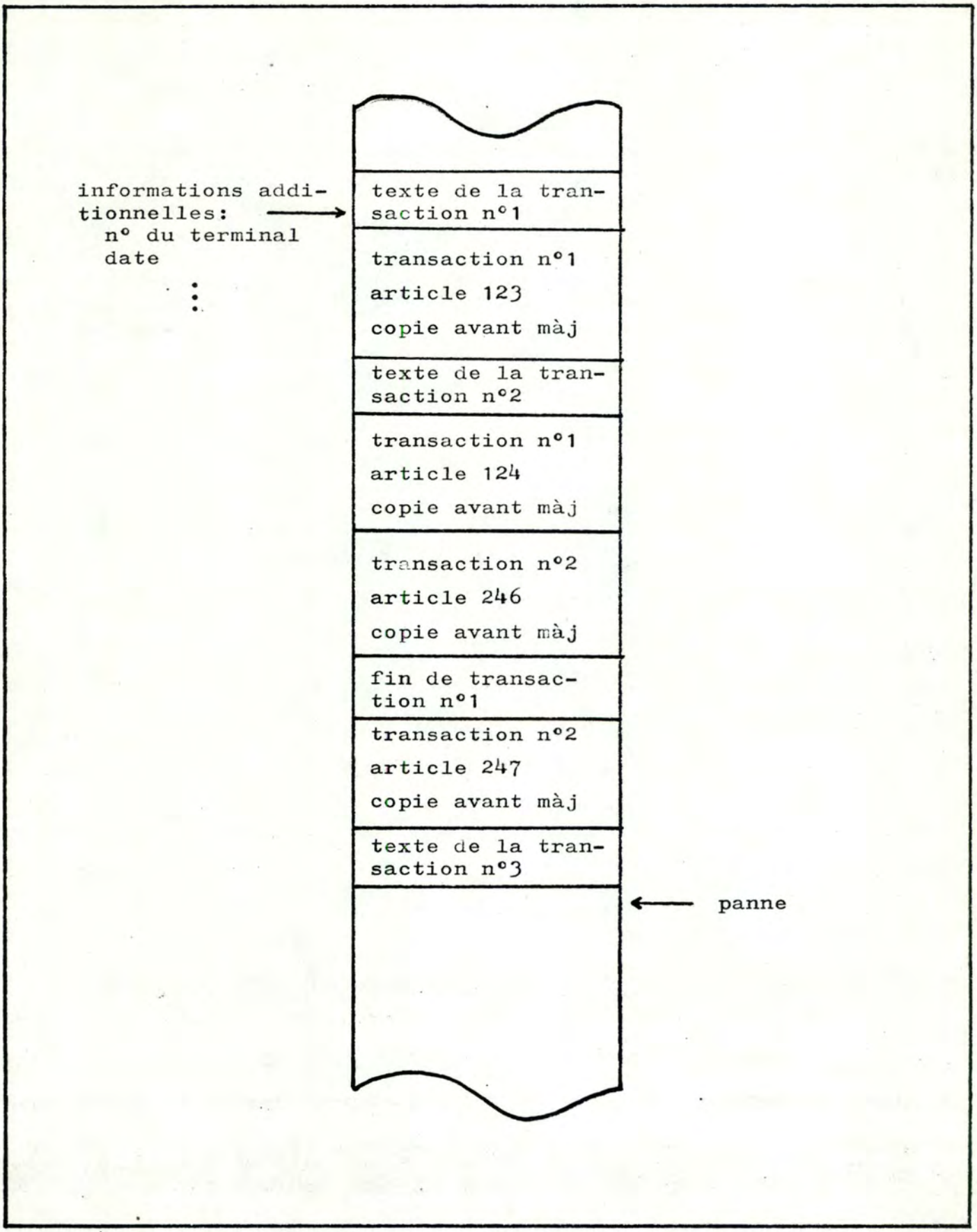

informations additionnelles:
n° du terminal
date
texte de la transaction n°1
transaction n°1
article 123
copie avant màj
texte de la transaction n°2
transaction n°1
article 124
copie avant màj
transaction n°2
article 246
copie avant màj
fin de transaction n°1
transaction n°2
article 247
copie avant màj
texte de la transaction n°3
panne

figure 2.8

audit trail afin d'éliminer toutes les transactions terminées. Il va utiliser la copie "before" des articles mis à jour par les transactions non complètement traitées pour restaurer la data base.

Le programme de recouvrement peut alors:

- recommencer l'exécution des transactions non achevées et la terminer.
- informer les utilisateurs de leur dernière transaction complète et leur demander de les réintroduire toutes à partir de celle-là.

En cas de destruction logique massive de la data base, (erreur programme constatée longtemps après) il faut charger le dernier dump avant l'erreur et recommencer le traitement de toutes les transactions entrées depuis lors, après correction de cette erreur. (cfr journaling)

Destruction physique.

La procédure est identique à celle du journaling. Chargement du dernier dump de la base de données et traitement de toutes les transactions subséquentes.

L'avantage de cette méthode est de ne recharger un dump complet qu'en cas de destruction logique massive ou de destruction physique des fichiers.(dommages les moins fréquents)

24.43 Texte de la transaction et copie des articles après màj.

Si la bande audit trail contient le texte de la transaction et la copie des articles après mise à jour, le schéma de recouvrement consiste à recharger le dernier dump de la base de données et à le fusionner avec l'audit trail en utilisant les copies "after" des transactions terminées afin de ne pas les retraiter.

Cette approche est efficace si le traitement d'une transaction prend beaucoup de temps CPU et requiert un montant énorme de calculs.

Elle est moins efficace si le rechargement de la data base dure longtemps.

En cas de destruction logique massive des fichiers, les contenus des articles n'étant pas corrects, la copie après mise à jour ne peut plus être utilisée. Il faut recharger le dernier dump et retraiter toutes les transactions après correction de l'erreur.

24.44 Texte de la transaction + copie de l'article avant mise à jour + copie de l'article après mise à jour. (figure 2.9)

Cette technique combine tous les avantages vus précédemment. La logique de recouvrement est identique à celle appliquée dans le cas des journaux.

Recouvrement.

Destruction logique.

- éliminer les transactions terminées.
- restaurer la data base avec les copies avant mise à jour et recommencer le traitement des autres transactions ou demander aux utilisateurs de les réintroduire.

Destruction physique.

- charger le dernier dump de la data base.
- fusionner le dump et l'audit trail en utilisant les copies after des articles, relatives aux transactions terminées.
- retraiter toutes les transactions non terminées, entrées depuis le dump.

Destruction logique massive.

- charger le dernier dump de la data base, précédent l'origine temporelle de l'erreur.
- recommencer le traitement de toutes les transactions subséquentes au dump.

NB. Dans toutes ces techniques, remarquons que les articles du fichier audit trail ne doivent pas être bloqués. Un accès physique équivaut donc à un accès logique.

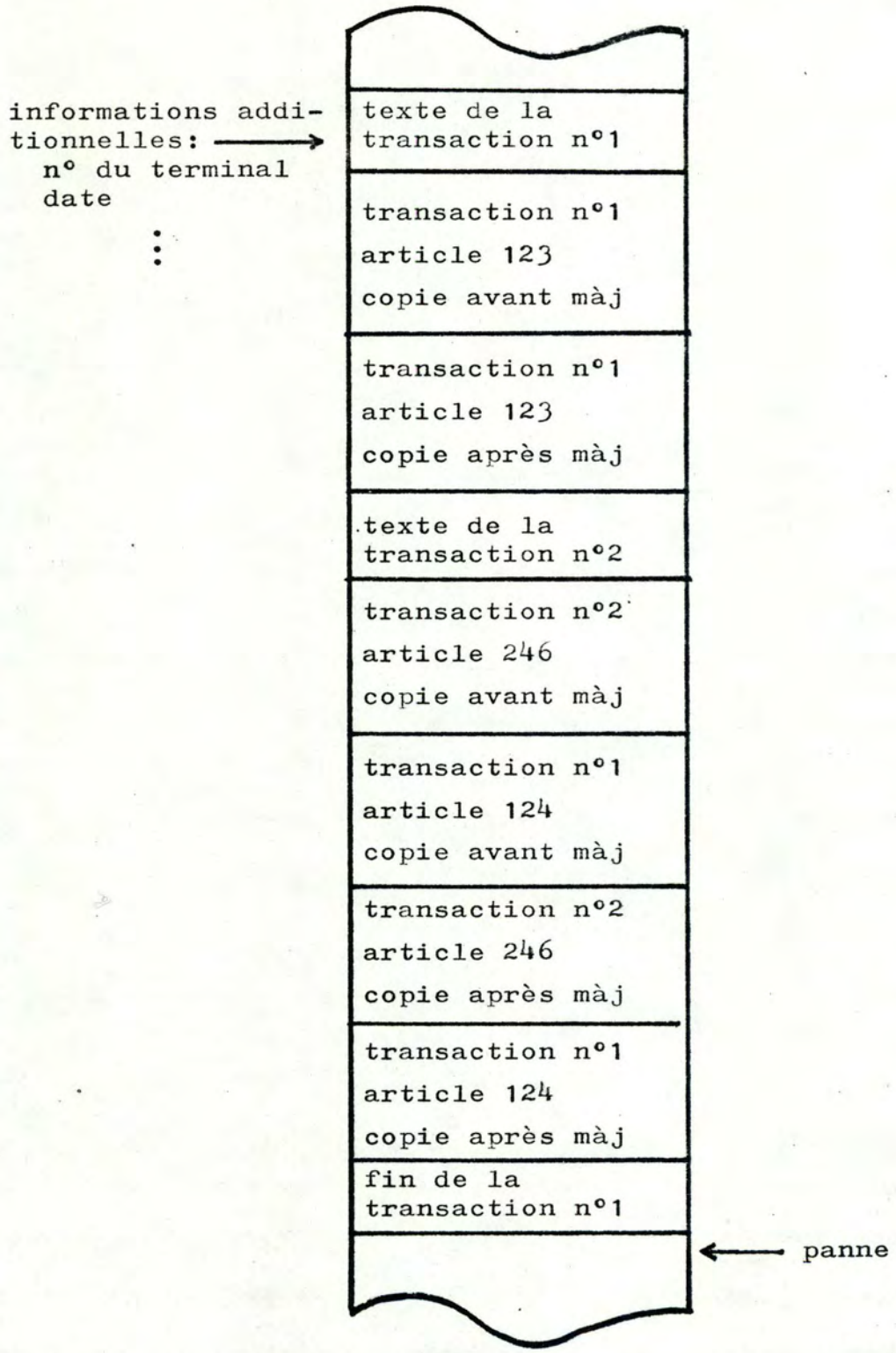

informations additionnelles:
n° du terminal
date
texte de la transaction n°1
transaction n°1
article 123
copie avant màj
transaction n°1
article 123
copie après màj
texte de la transaction n°2
transaction n°2
article 246
copie avant màj
transaction n°1
article 124
copie avant màj
transaction n°2
article 246
copie après màj
transaction n°1
article 124
copie après màj
fin de la transaction n°1
panne

figure 2.9

En bloquant les articles, une panne engendrant la perte du contenu de toute ou d'une partie de la mémoire pourrait impliquer la perte de transaction pour l'audit trail et des copies des articles avant et après mise à jour.

Messages aux utilisateurs.

Informer l'utilisateur sur le statut de sa transaction après une panne est une tâche aisée car l'audit trail fournit les renseignements nécessaires pour le faire (texte de la transaction, numéro unique par transaction, numéro du terminal,...).

Il peut arriver que le système tombe en panne entre l'envoi de la transaction par l'utilisateur et sa réception par le système. Celui-ci ne peut donner aucun message de bonne ou mauvaise réception à l'utilisateur dès que la panne est réparée.

La meilleure approche est d'envoyer à l'utilisateur une reconnaissance positive pour chaque transaction terminée. Elle peut être très simple (OK p.e), mais elle peut aussi contenir le numéro de transaction comme référence en cas de panne.

Exemple:

Commande 1234 modèle 13 bleu
ØK N° 31335
Commande 1235 modèle 18 rouge
ØK N° 31347

Remarque:

Les numéros de transactions venant du système sont croissants mais pas nécessairement consécutifs eu égard de l'interpénétration avec les autres utilisateurs travaillant sur les mêmes programmes et données.

Le programme de recouvrement scanne la bande audit trail et construit une table contenant le numéro de la dernière transaction correctement terminée, pour

chaque utilisateur en contact avec le système au moment de la panne. La table est utilisée au redémarrage pour engendrer un message du type:

LAST VALID TRANSACTION WAS N° 31347 AT 1:28 PM.

Trois possibilités:

1) l'utilisateur a envoyé une transaction mais n'a pas reçu de reconnaissance de bonne ou mauvaise réception par le système, àson terminal. Si le système n'a pas pu traiter le message avant la panne, la dernière transaction valable refère à une transaction précédente.
2) situation identique à celle ci-dessus, mais le système a enregistré la transaction sur le ruban audit trail et l'a complètement traitée. Il n'a cependant pas eu le temps d'envoyer une reconnaissance positive.
   Le "last valid transaction" porte ici un numéro supérieur à n'importe lequel déjà écrit au terminal.
3) l'utilisateur a envoyé un message, reçu une réponse positive et il n'a pas retourné d'autres messages.
   La dernière transaction valable refère à la dernière transaction envoyée.

Le numéro de la transaction peut être envoyé une première fois par le système dès la réception. Si l'utilisateur ne le reçoit pas, il devra retransmettre cette transaction, sinon il s'en abstiendra sauf si le système le lui demande.

Quand on construit un schéma de recouvrement de fichier, il est important de se souvenir que le système peut tomber en panne à n'importe quel point du traitement de la transaction. Beaucoup d'ennuis viennent de la non prise en considération des endroits où la probabilité de panne est faible.

Etant donné le volume des traitements qui s'exé-

cutent dans la plupart des systèmes, il devient très probable qu'une panne arrive en un point critique après une semaine d'exploitation ou un mois.

Endroits du traitement où une panne peut survenir.

1) L'utilisateur imprime sa transaction sur le terminal et pousse sur la"transmit key". Une panne à cet endroit n'implique aucun dommage pour la base de données car l'input n'a pas encore été reconnu par le système.
L'utilisateur peut être informé de statut de sa transaction comme indiqué ci-avant.

2) Le système lit l'input, mais le programme application ne l'a pas encore pris en charge.
La data base n'est pas endommagée.

3) Le programme application prend la transaction en charge, l'écrit dans le fichier audit trail avec un numéro d'utilisateur, un numéro de terminal, un numéro de transaction, la date, l'heure, etc...
La base de données n'a pas encore été affectée, mais le programme de recouvrement tiendra compte de cette transaction. Il pourra soit la retraiter durant sa phase d'exécution ou demander à l'utilisateur de la retransmettre.

4) Quand le programme application est prêt au traitement, il lit certains articles de la data base.
La situation est identique à celle du point 3.

5) Avant de mettre à jour un ou plusieurs articles, le programme application en écrit d'abord une copie courante ou copie "before" dans le fichier audit trail.
Il n'y a pas encore eu de mise à jour dans la data base mais le programme de recouvrement utilisera en cas de

panne utilisera la copie before pour effectuer la restauration.

6) Le programme application met à jour des articles de la base de données.
Une panne ici entraînera probablement une perturbation de la data base. L'audit trail permettra au programme de recouvrement de la restaurer.
NB. Le programme application doit écrire la copie before et en attendre l'achèvement avant de mettre à jour la data base.

7) Quand le programme application a terminé une transaction, il l'indique en écrivant un enregistrement spécial dans le fichier audit trail (fin de transaction).
En cas de panne avant l'écriture de la fin de transaction, le programme de recouvrement restaure la data base et la transaction sera retraitée bien qu'elle fût correctement terminée.
Si la panne arrive après, le programme de recouvrement reconnaîtra que la transaction est terminée, elle ne sera pas retraitée.

8) Un message de fin de transaction est imprimé sur le terminal de l'utilisateur. Il sait alors que sa transaction est correctement accomplie.

24.5 CHECKPOINT.

Enregistrement périodique du statut du système et des fichiers sur une bande magnétique ou un DASD. Cet enregistrement peu être fait à des ruptures logiques des jobs, mais le critère le plus communément utilisé dans les systèmes de télétraitement est un nombre prédéterminé de transactions ou une interruption d'horloge.

Exigences du checkpoint:

- quelles informations retenir?
- quelles sont celles qui vont composer l'enregistrement checkpoint?
- sur quelle base de temps prendre les checkpoints?
- combien de temps peut on consacrer au restart?

Certaines applications ne recourent pas au checkpoint, d'autres accordent beaucoup d'importance à un restart exact.

La disponibilité CPU, le temps canal, la compétence des programmes, l'aisance à reprendre les transactions entrées, l'intégrité des données influencent la technique du checkpoint.

Les procédures de recouvrement doivent être efficaces et simples à traiter.

Dans un système de télétraitement, on utilise le checkpoint conjointement avec le journaling. On garde un fichier journal des transactions et un fichier journal enregistrant les copies avant mise à jour des articles de la base de données.

Le checkpoint ne concerne pas les fichiers de données, mais bien les deux fichiers journaux.

L'utilité de cette technique est d'éviter la réexécution de toutes les transactions enregistrées dans

le journal des transactions et de permettre la restauration de la data base à partir de ce point.

Remarque.

Que le système travaille en batch ou en on-line, d'un point de vue fichier, le checkpoint n'a sa raison d'être qu'avac l'existence d'un moyen de restauration des fichiers. Il permet l'amélioration de ce moyen.
Exemple:

on-line: journaling ou audit trail + checkpoint de ces fichiers. La restauration s'effectue jusqu'au checkpoint et la réexécution des transactions commence en ce point.

batch: grandfather, father, son + checkpoint des programmes. Le traitement ne recommence pas au début, mais à partir du checkpoint, le fichier père garantissant une data base exacte.

La technique du journaling + checkpoint est l'objet de la troisième partie.

PARTIE 3

DEVELOPPEMENT D'UNE METHODE DE RECOUVREMENT DES FICHIERS

31 BUT DE L'ETUDE

Dans un système de télétraitement, les messa-entrés sont sauvés dans des fichiers adéquats et également placés dans des files d'attente.

Pour son exécution, un message a besoin de un ou plusieurs programmes applications spécialisés à lire écrire, traiter et mettre à jour des données figurant dans des fichiers applications (base de données). Un message peut mettre à jour des données dans plusieurs de ces fichiers.

A un moment tout à fait quelconque, une panne peut survenir, dans l'alimentation en électricité par exemple, laissant les utilisateurs ignorants:

1- des données effectivement mises à jour, de celles en train de l'être, ou des autres ne l'étant pas encore.

2- des messages entièrement traités et de celui ou ceux en cours d'exécution au moment de la panne.

Il importe donc d'utiliser un moyen permettant de soulever ces deux incertitudes et de redémarrer l'exploitation avec une banque de données correcte, et dans le meilleur des cas au début des messages en exécution au moment de la panne.

Dans cette optique, nous développerons un système de recouvrement utilisant un fichier des transactions (ou messages), un fichier de sauvetage des articles de la data base avant mise à jour et prenant un checkpoint périodique des fichiers de sauvetage.

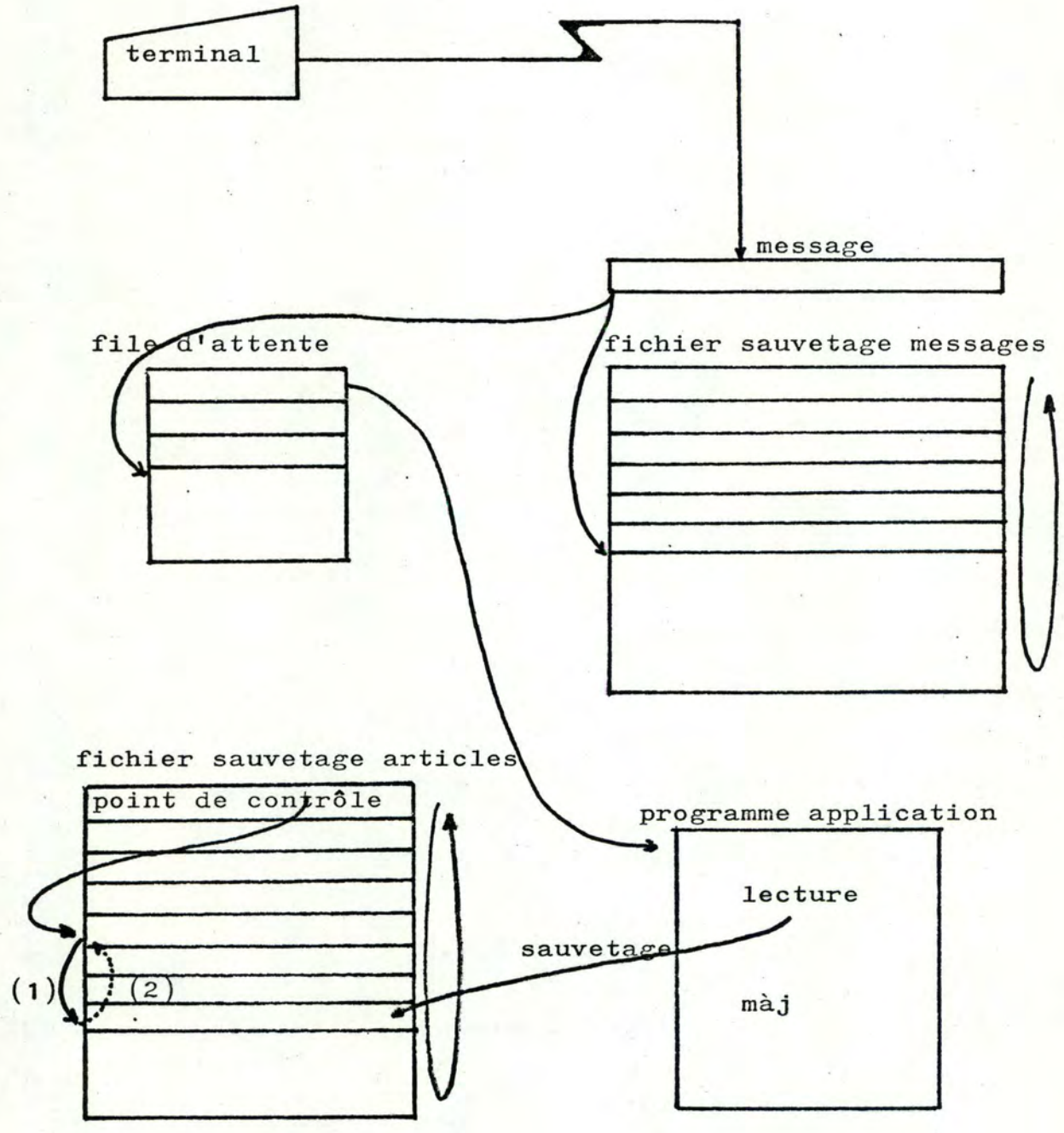
32 CONCEPTION GENERALE DU SYSTEME

Des messages sont envoyés par terminaux éloignés vers le centre de traitement.(figure 3.1) Ils sont enregistrés dès leur arrivée dans un fichier de sauvetage des messages (transaction journal). Ils sont ensuite analysés, et selon leur type, une programme spécialisé effectue un certains nombre d'opérations dans la base de données.

Chaque article, avant d'être mis à jour, est copié dans un fichier de sauvetage des articles (file actions journal: copie before). On y enregistre également des informations supplémentaires de localisation de ces articles dans la data base. La mise à jour est effectuée.

Périodiquement, le système prend un checkpoint des fichiers de sauvetage. Ce checkpoint permet un recouvrement aisé des fichiers en cas de panne dans l'installation et il garantit une mise à jour correcte et complète de tous les articles qui le précèdent.

Le chapitre 33 est l'objet d'un développement en détail de cette méthode.



(1) balayage
(2) restauration

figure 3.1

33 DEVELOPPEMENT DE LA METHODE

33.1 SYSTEME GLOBAL (sauf la restauration). figure 3.2 a et b

Des cartes perforées simulent l'envoi des messages par les terminaux. Elles constituent une file d'attente exploitée d'une manière FIFO (first in, first out). Le système affecte au message un numéro de séquence qui lui est propre (numéro de séquence interne) afin de différencier les messages portant un même numéro de séquence donné par les utilisateurs (ceux-ci ajoutant un numéro de séquence propre à chacune de leurs transactions, plusieurs d'entre elles, venant de terminaux différents peuvent porter un numéro de séquence identique) et pour établir une correspondance entre la transaction et les articles qu'elle met à jour. Les messages (transactions) sont ensuite sauvés dans un fichier de sauvetage des messages avec les numéros de séquence interne et externe. Nous appelons numéro de séquence externe celui donné par l'utilisateur.

Nous utilisons deux fichiers de sauvetage des messages afin de pouvoir tirer une copie de l'un pendant que l'autre est en action. Un switch - SW1 - indique celui qui est en cours d'utilisation.

Une analyse du code opération figurant dans le message permet de céder la main à un programme application. Nous avons écrit trois programmes de ce type répondant aux codes 0001, 0002 et 0003. Ils sont réduits à des lectures et des écritures de mise à jour dans les fichiers applications car notre but n'est pas de les développer en détail.

Principe fondamental:

- une lecture seule n'engendre aucun sauvetage.
- une lecture suivie d'une mise à jour donne lieu à un

sauvetage de l'article lu.
- une création est précédée d'une lecture et du sauvetage de la zone dans laquelle l'article va être placé.
- une suppression est précédée de la lecture de l'article à supprimer et du sauvetage de celui-ci.

Après chaque lecture (sauf lecture seule) la main est donnée à un programme de sauvetage des articles des fichiers applications.

Les articles, avant mise à jour sont écrits dans un fichier de sauvetage. Nous utilisons en fait deux fichiers de ce type pour les mêmes raisons que celles qui nous ont incités à prendre deux fichiers de sauvetage des messages.

Sont mémorisés:

- les numéros de séquence interne et externe.
- le nom du fichier application contenant l'article sauvé.
- l'adresse dans le fichier application de l'article sauvé.
- le contenu de l'article.

Ces informations nous permettrons de restaurer aisément les fichiers applications après une panne.
Grâce au numéro de séquence interne, onétablit une correspondance entre le message et l'article sauvé. Si le message a entraîné la mise à jour de plusieurs articles, ceux-ci auront le même numéro de séquence interne, celui du message. Le programme de sauvetage conserve aussi une zone contenant les informations qui constitueront l'enregistrement checkpoint. (voit ci-dessous)

Le programme application reprend la main pour terminer l'exécution de la transaction et il la repasse ensuite au programme principal qui va tester s'il faut prendre un checkpoint.

OUI: en tête du fichier de sauvetage des articles, on enregistre:

- les numéros de séquence interne et externe

- le nom du fichier de sauvetage dans lequel l'article est sauvé
- l'adresse dans ce fichier de sauvetage de l'article sauvé.

Ces informations sont contenues dans une zone gérée par le programme de sauvetage. Celui-ci la met à jour à chaque écriture d'article dans les fichiers de sauvetage.

Une fois l'écriture du checkpoint terminée le système prend en charge le message suivant.

NON: lecture du message suivant.

33.2 PROGRAMMATION.

33.21 Enchaînement des programmes.

La figure 3.3 nous donne une idée générale de l'enchaînement des différents programmes.

Le début du traitement est effectué par le programme principal. En fonction du code opération, celui-ci cède la main à un des trois programmes applications. De ces programmes applications, nous passons ensuite après chaque lecture au programme de sauvetage. Celui-ci rend la main pour la mise àjour au programme application qui la repasse ensuite au programme principal.

Le programme principal appelle le programme de checkpoint si nécessaire et il prend en charge le message suivant.

Nomenclature:

- programme principal: CMAIN
- programmes applications: CSECT2, CSECT3, CSECT4
- programme de sauvetage: CSAUV
- programme de checkpoint: CKPT

33.22 Programme principal. figure 3.4 a et b

Comme nous l'avons déjà vu, un numéro de séquen-

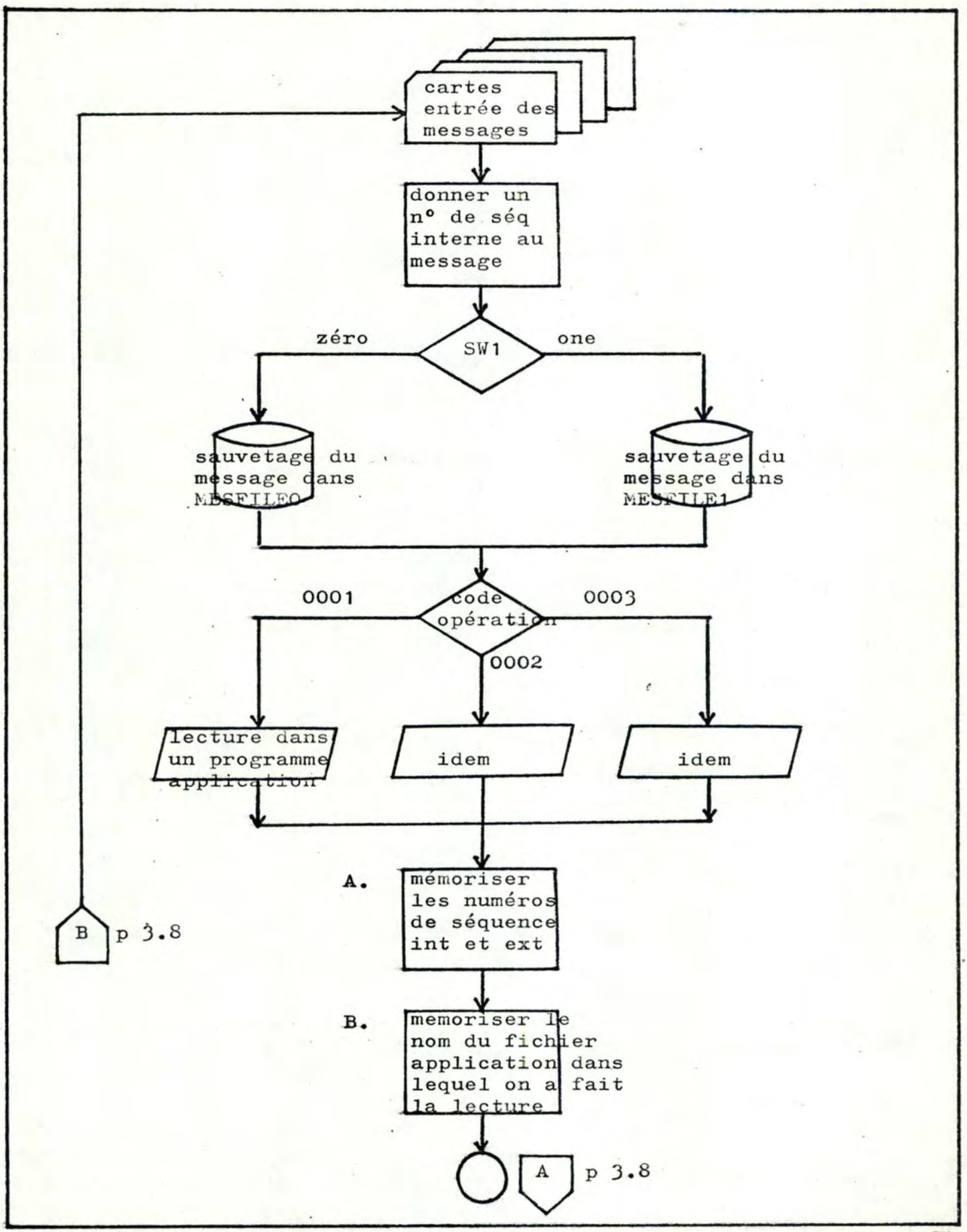

cartes entrée des messages
donner un n° de séq interne au message
zéro
SW1
one
sauvetage du message dans MESFILE0
sauvetage du message dans MESFILE1
0001
code opération
0003
0002
lecture dans un programme application
idem
idem
A.
mémoriser les numéros de séquence int et ext
B.
memoriser le nom du fichier application dans lequel on a fait la lecture
B p 3.8
A p 3.8

Figure 3.2a

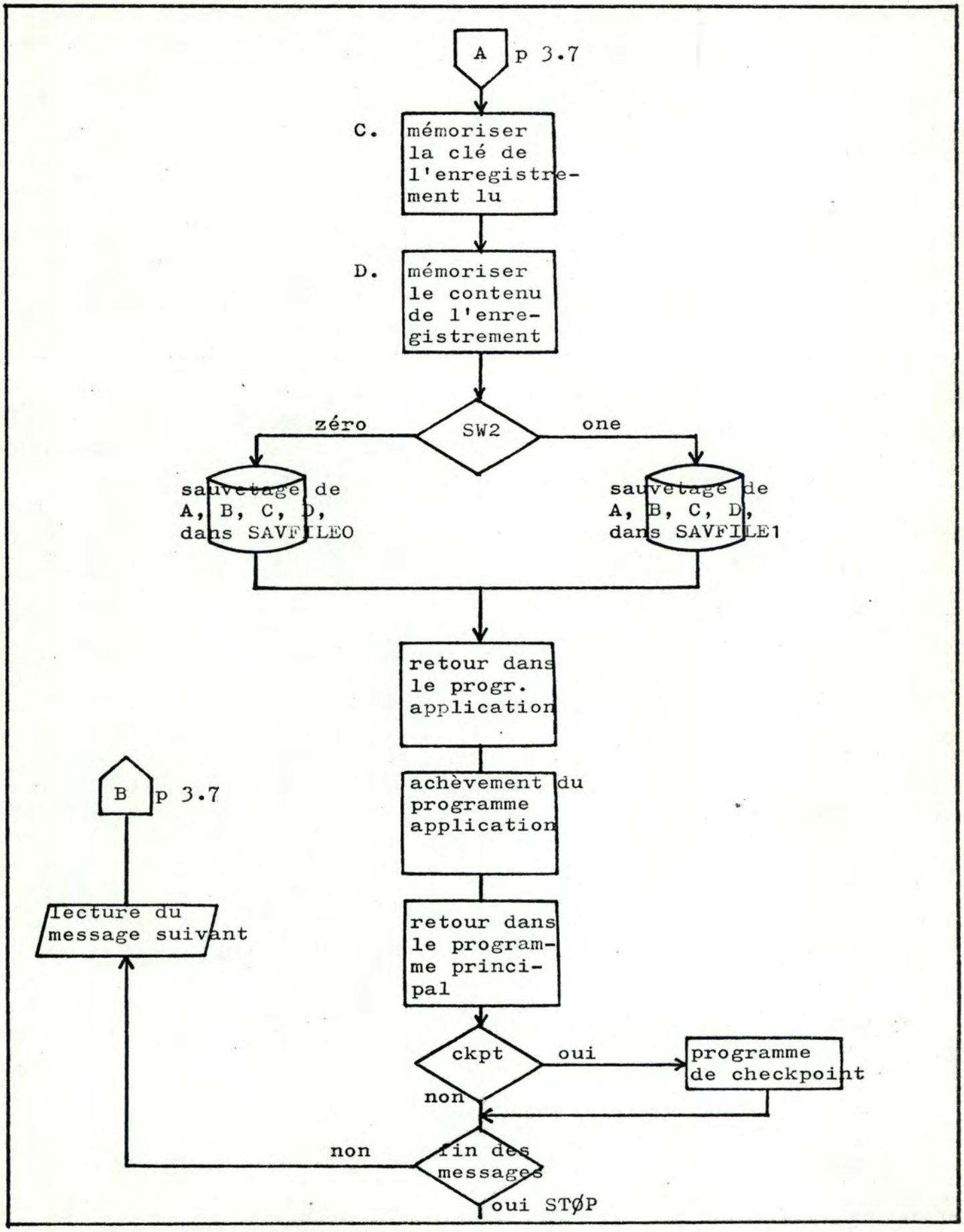

A
p 3.7
C.
mémoriser la clé de l'enregistrement lu
D.
mémoriser le contenu de l'enregistrement
SW2
zéro
one
sauvetage de A, B, C, D, dans SAVFILE0
sauvetage de A, B, C, D, dans SAVFILE1
retour dans le progr. application
achèvement du programme application
retour dans le programme principal
ckpt
oui
programme de checkpoint
non
fin des messages
non
oui STØP
B
p 3.7
lecture du message suivant

figure 3.2 b

enchaînement des programmes: sauf la restauration.

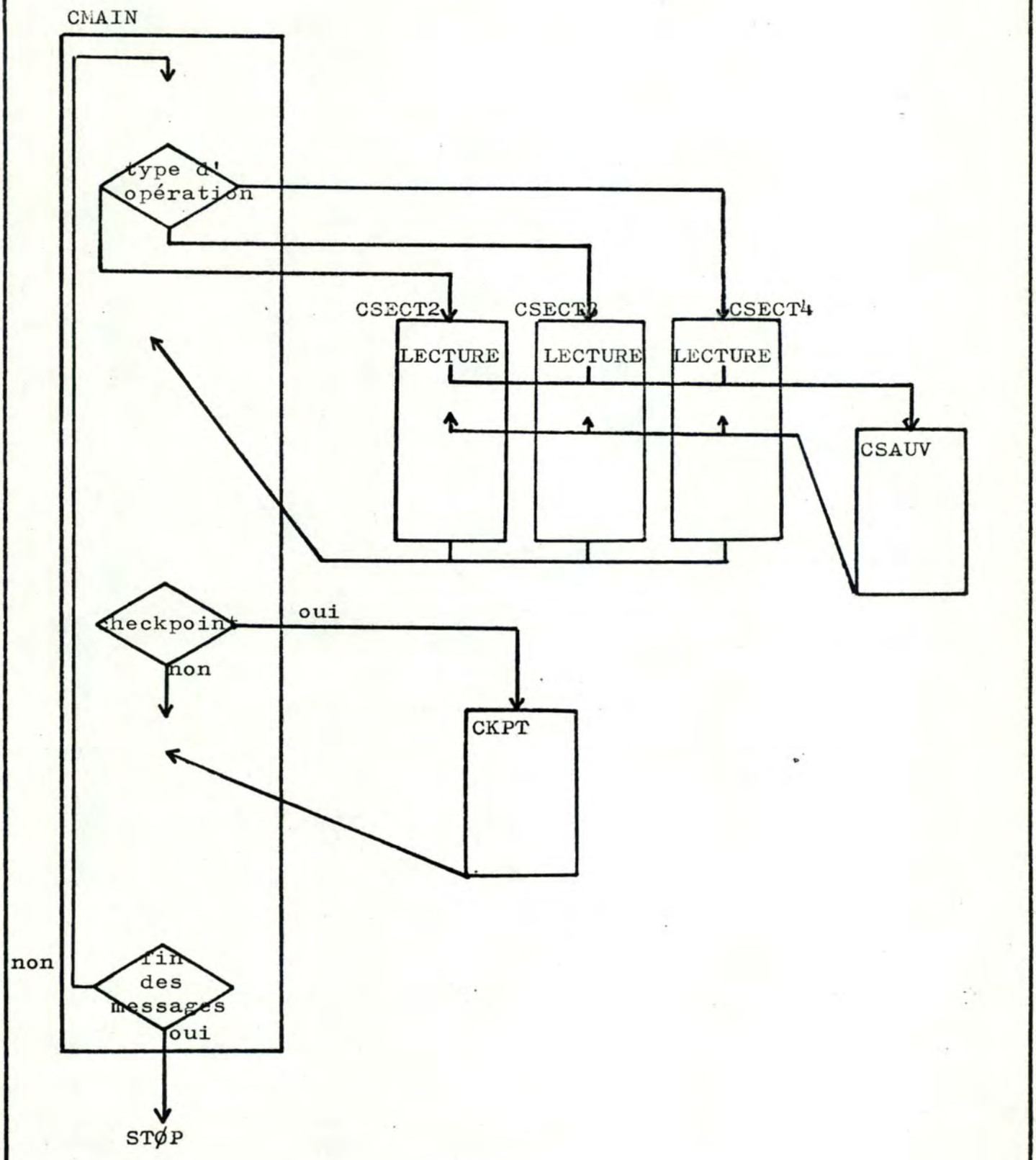


figure 3.3

ce interne est ajouté au message et celui-ci est ensuite écrit sur un des fichiers de sauvetage des messages. Ces deux fichiers sont appelés MESFILE0 et MESFILE1. Ils sont gérés en accès direct relativement au début du fichier (relative direct access method). La clé doit être fournie en binaire sur trois caractères et être comprise entre 0 et M (M = nombre maximum d'articles du fichier - 1).

Exemple:

100 articles au maximum dans le fichier.
les clés varient de 0 à 99.
Nous trouvons dans la figure 3.5 un dessin du format des articles de MESFILE0 et MESFILE1.

Grâce à cette technique d'accès, nous pouvons établir une correspondance directe entre le numéro de séquence interne et la clé d'enregistrement dans les fichiers de sauvetage des messages, dont nous avons limité la taille à 100 articles.

$$\frac{\text{n° de séq int}}{100} + \text{reste}$$

Le reste est pris comme clé d'accès. Cet artifice évite la prise d'un checkpoint de ces fichiers. Au redémarrage, le numéro de séquence interne étant mémorisé dans le checkpoint des fichiers de sauvetages des articles avant mise à jour, on effectue le même calcul pour accéder directement au message correspondant au point de contrôle. Nous pourrons dès lors avertir les utilisateurs aux terminaux de leur dernier message correctement traité.

L'utilisation de deux fichiers nous permet de tirer une copie de l'un pendant que nous travaillons sur l'autre. Avant chaque écriture, nous testons un switch pour déterminer quel fichier est en cours d'utilisation.
SW1 = 0: MESFILE0 est utilisé.
SW1 = 1: MESFILE1 est utilisé.

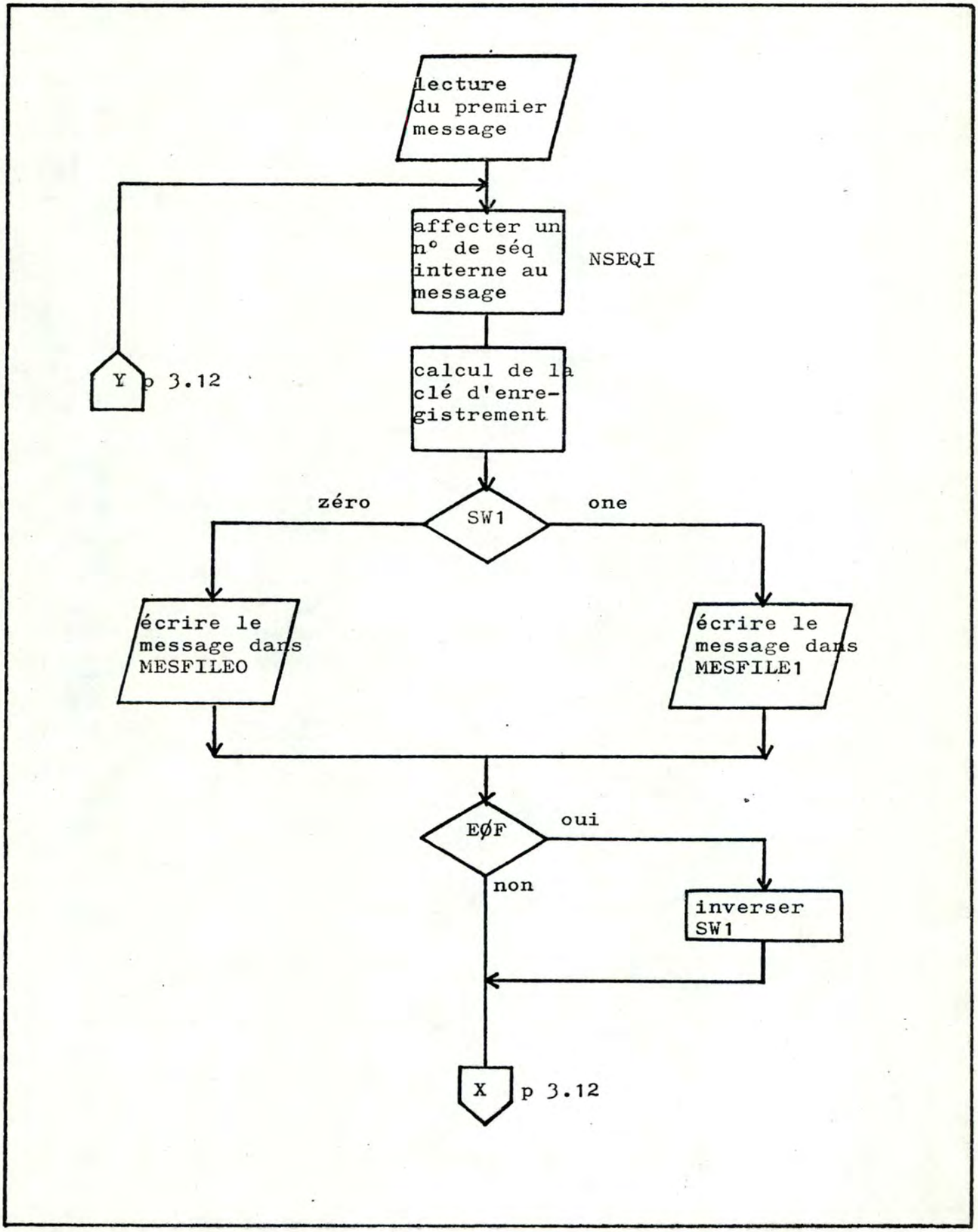

lecture du premier message
affecter un n° de séq interne au message
NSEQI
calcul de la clé d'enregistrement
Y p 3.12
zéro
SW1
one
écrire le message dans MESFILEO
écrire le message dans MESFILE1
EØF
oui
non
inverser SW1
X p 3.12

figure 3.4 a

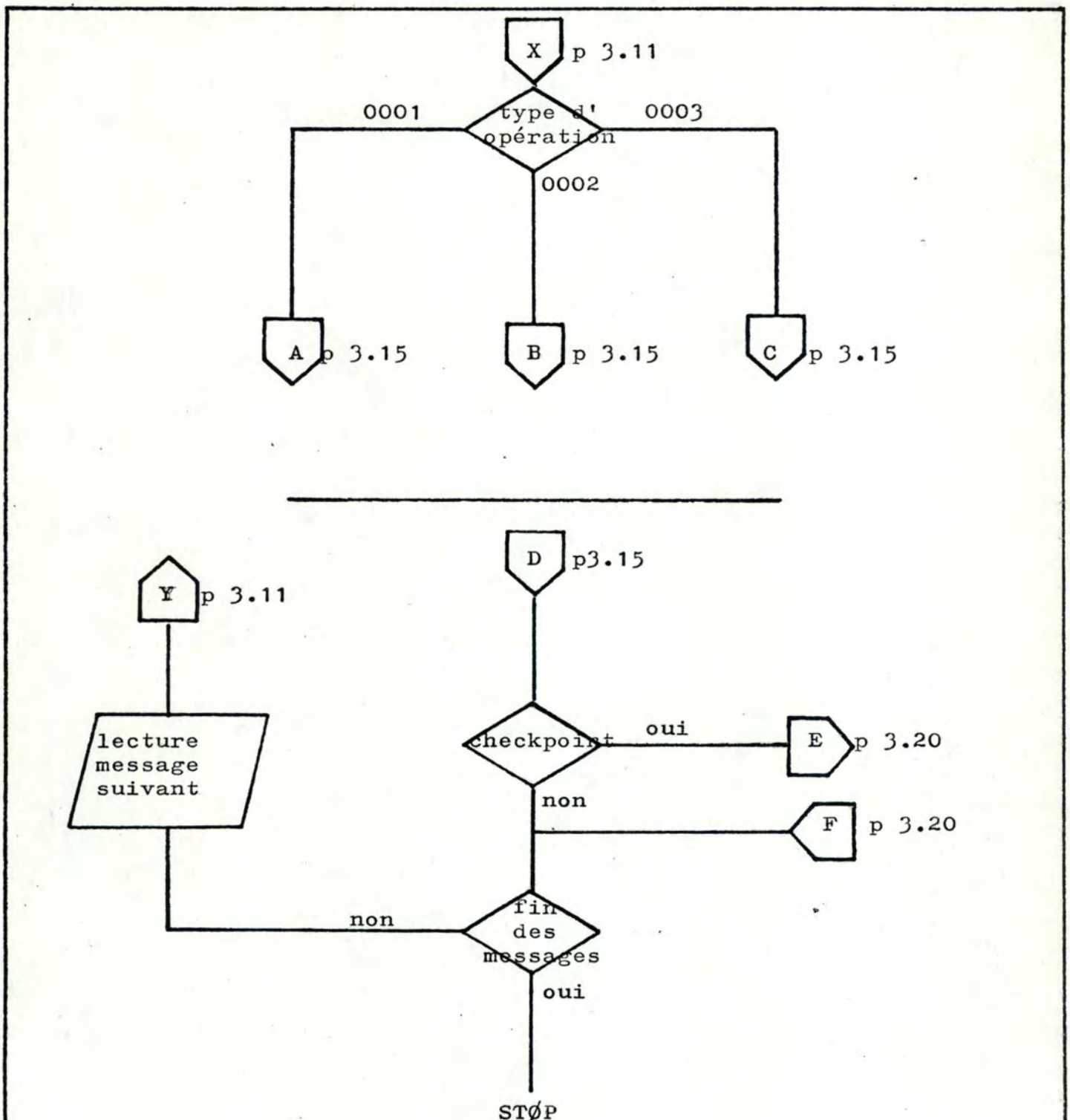

X p 3.11
0001
type d' opération
0003
0002
A p 3.15
B p 3.15
C p 3.15
Y p 3.11
D p3.15
lecture message suivant
checkpoint
oui
E p 3.20
non
F p 3.20
non
fin des messages
oui
STØP

figure 3.4 b

Une fois un fichier rempli, nous inversons le switch pour passer à l'autre.

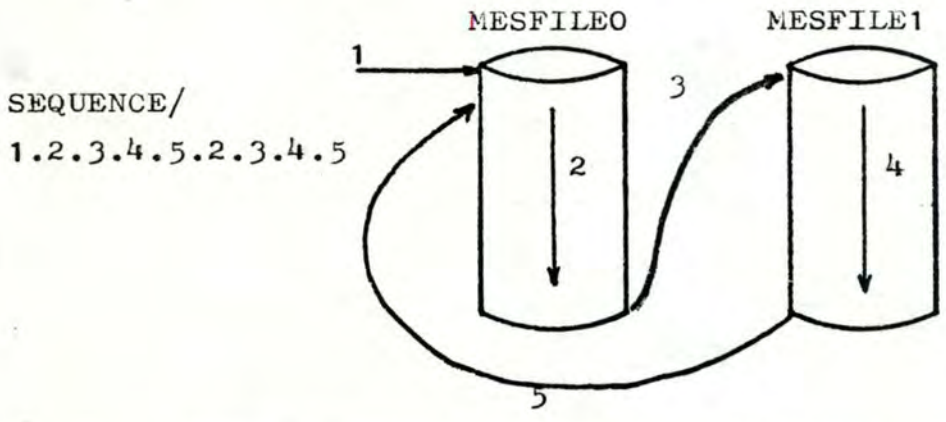


figure 3.5

Nous trouvons dans la figure 3.6 un dessin du format des articles de MESFILE0 et MESFILE1.

33.23 Programmes applications.

Les trois programmes applications que nous avons écrits travaillent sur deux fichiers applications: APFILE10 et APFILE11. Ces programmes effectuent des lectures/écritures et appellent la routine de sauvetage (CSAUV) après chaque lecture.

La figure 3.7 explique la logique de ces programmes.

33.24 Programme de sauvetage. figure 3.8 a et b

Quand un programme application lit un article avant une mise à jour, il appelle une routine qui va en sauver le contenu avant modification.

Celle-ci mémorise le numéro de séquence interne afin de déterminer de manière unique le message ayant engendré la lecture et la mise à jour (nous avons retenu les numérosde séquence interne et externe).

Dessin des articles des fichiers MESFILE0 et MESFILE1.

4	4	4	4	30
NSEQI	NSEQE	NTERM	TOP	ZQCQ

Longueur totale = 46 caractères
NSEQI = n° de séquence interne en binaire sur 4 caractères
NSEQE = n° de séquence externe donné par l'utilisateur
NTERM = n° d'identification du terminal
TOP = type d'opération: 0001, 0002, 0003
ZQCQ = zone quelconque indéfinie dans notre système

Dessin des articles des fichiers SAVFILE0 et SAVFILE1.

4	4	14	3	40
NSEQI	NSEQE	FILENAME	CLE	CONTENU

Longueur totale = 65 caractères
NSEQI et NSEQE = voir ci-dessus
FILENAME = nom du fichier applications contenant l'article sauvé avant mise à jour
CLE = clé d'accès dans ce fichier applications
CONTENU = contenu de l'article sauvé

NB. Pour des raisons pratiques, le nom de chaque fichier que nous utilisons comporte 14 caractères.
ex: JAMIN.SAVFILE0

figure 3.6

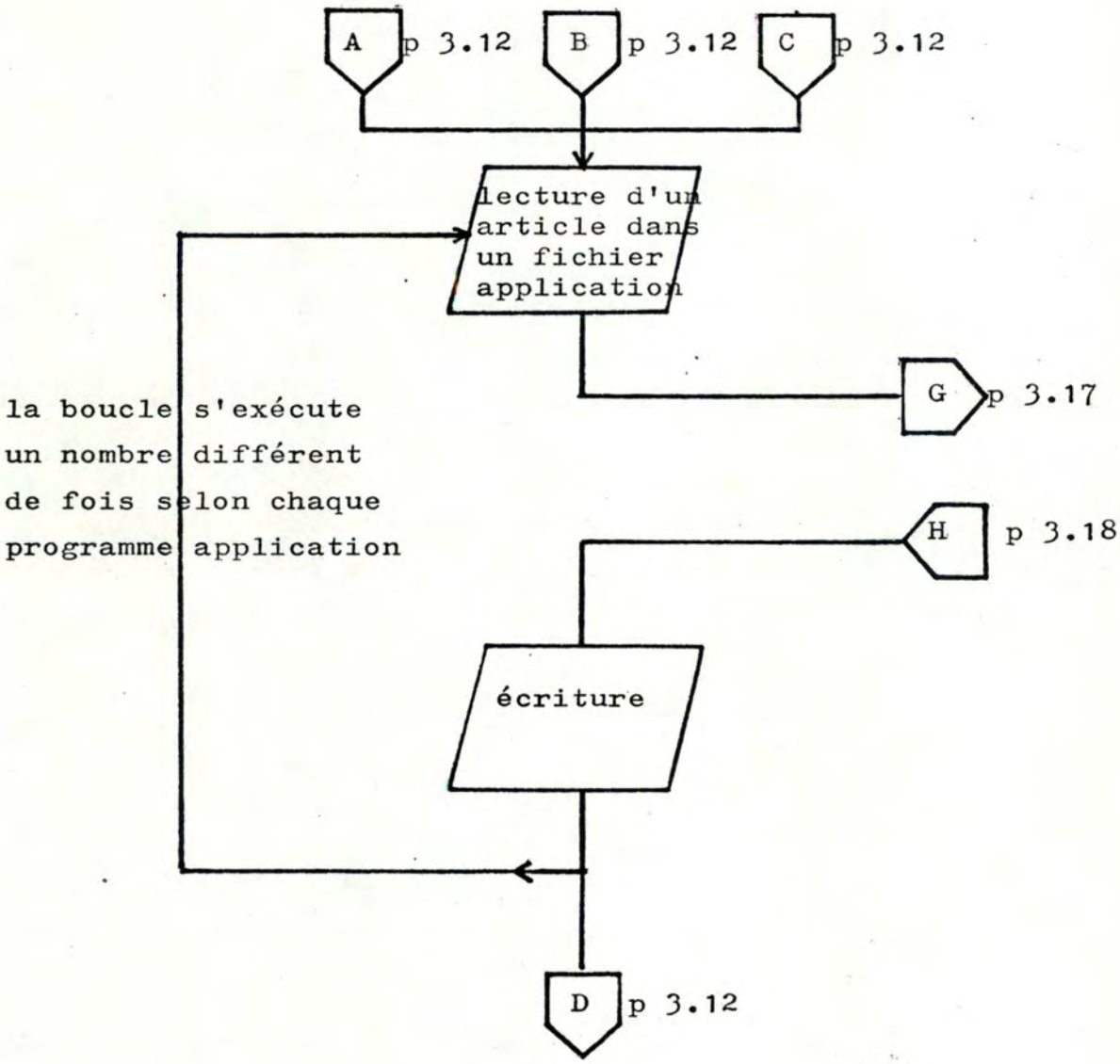

A p 3.12
B p 3.12
C p 3.12
lecture d'un article dans un fichier application
G p 3.17
la boucle s'exécute un nombre différent de fois selon chaque programme application
H p 3.18
écriture
D p 3.12

figure 3.7

Le système de télétraitement travaille sur un ensemble de fichiers applications constituant la base de données. Il est donc fondamental de mémoriser le nom du fichier contenant l'article sauvé afin de le restaurer au bon endroit en cas de panne.

Nous mémorisons aussi l'adresse de l'article dans le fichier application afin de le situer exactement lors de la restauration. Nous enregistrons finalement le contenu de l'article à modifier.

Le programme de sauvetage utilise deux fichiers pour les mêmes raisons que ci-dessus. Nous appelons ces deux fichiers SAVFILE0 et SAVFILE1. Un switch indique celui qui est en cours d'utilisation.

Structure des deux fichiers de sauvetage.

- articles de longueur fixe, non bloqués.
- les deux premiers articles de SAVFILE0 sont réservés au checkpoint (cfr 33.25)
- la logique d'écriture dans ces fichiers est la même que celle utilisée pour MESFILE0 et MESFILE1.

L'accès se fait directement, relativement au début des fichiers. La clé doit être fournie en binaire sur trois caractères.

Le programme de sauvetage gère une zone utilisée par le programme de checkpoint. Elle contient les numéros de séquence interne et externe du message, le nom du fichier dans lequel l'article vient d'être sauvé et l'adresse dans ce fichier.

NB. La figure 3.6 contient le dessin du format des articles de SAVFILE0 et SAVFILE1.

Articles non bloqués.

En décidant de bloquer les articles à sauver, il faut attendre le remplissage du buffer pour exécuter l'accès physique. Une panne détruisant le contenu de la mémoire engendrera la perte de message et d'articles avant mise à jour.

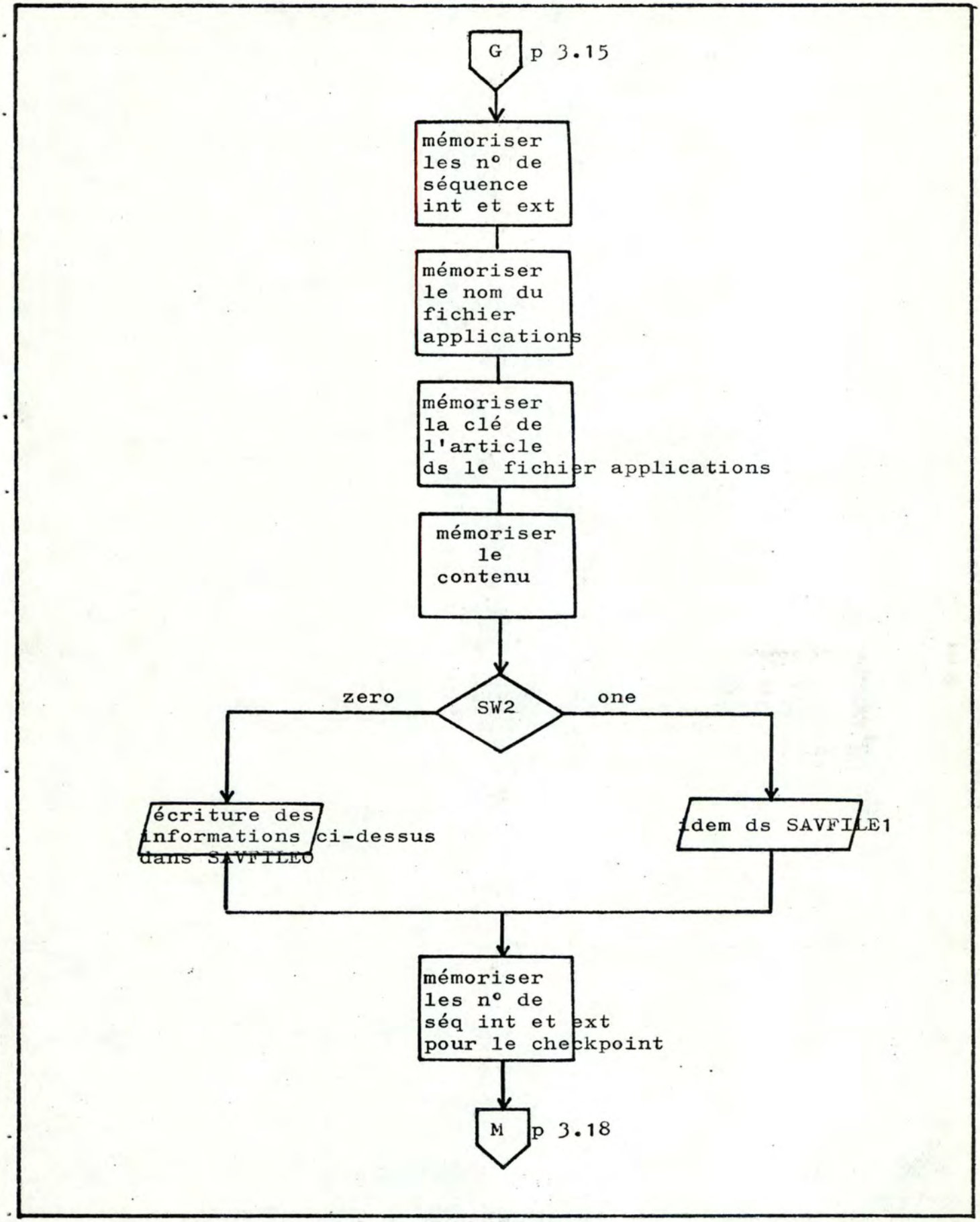

G
p 3.15
mémoriser les n° de séquence int et ext
mémoriser le nom du fichier applications
mémoriser la clé de l'article ds le fichier applications
mémoriser le contenu
zero
SW2
one
écriture des informations ci-dessus dans SAVFILE0
idem ds SAVFILE1
mémoriser les n° de séq int et ext pour le checkpoint
M
p 3.18

figure 3.8 a

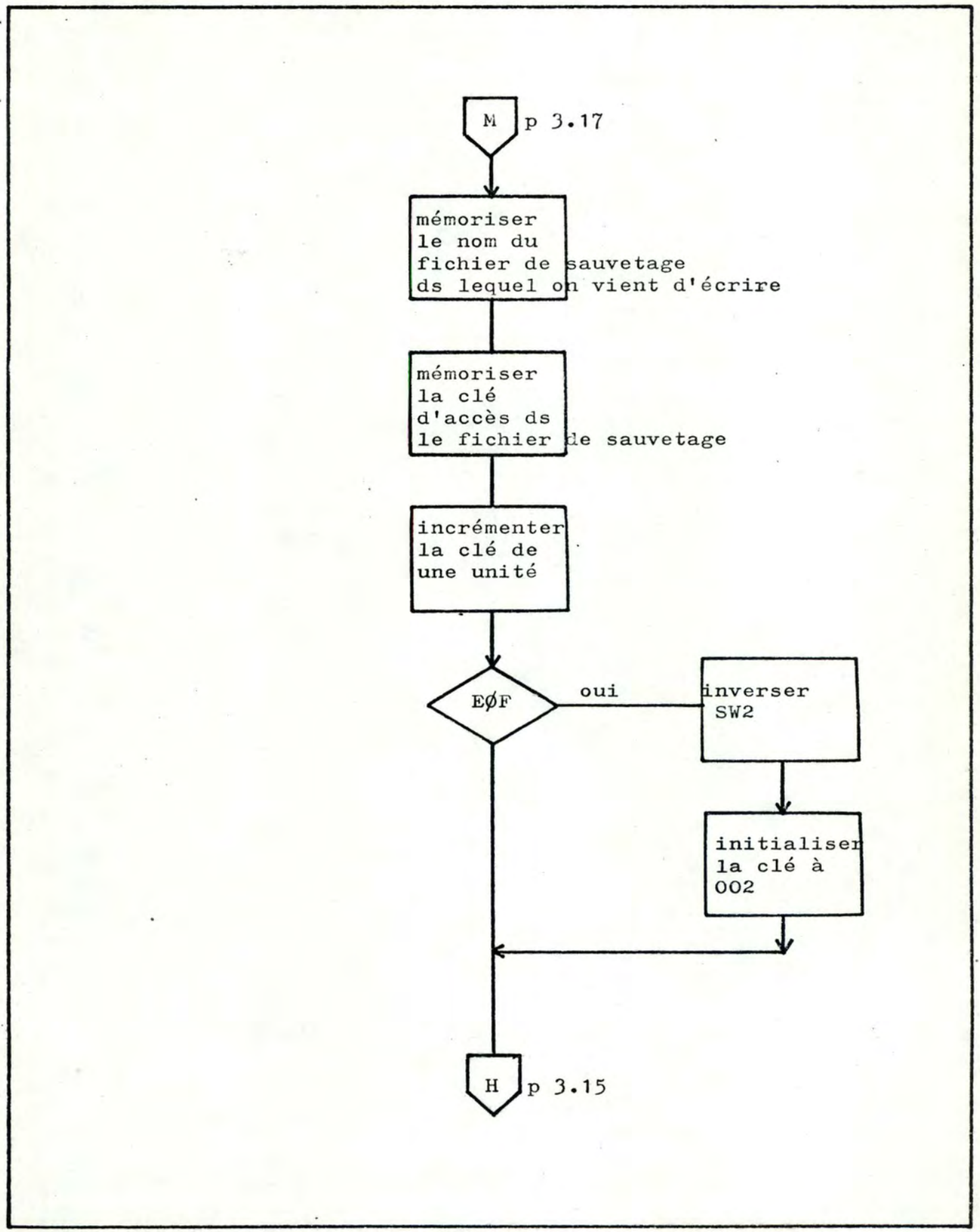

M
p 3.17
mémoriser le nom du fichier de sauvetage ds lequel on vient d'écrire
mémoriser la clé d'accès ds le fichier de sauvetage
incrémenter la clé de une unité
EØF
oui
inverser SW2
initialiser la clé à 002
H
p 3.15

figure 3.8 b

33.25 Programme de checkpoint. figure 3.9

Le programme de sauvetage met à jour une zone qui est écrite en début du premier fichier de sauvetage lors de la prise d'un checkpoint par le programme ad hoc. Cette zone contient:

- les n° de séquence interne et externe.
- le nom du fichier de sauvetage contenant l'article correspondant.
- l'adresse dans le fichier de sauvetage de l'article correspondant.

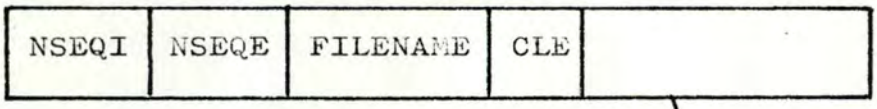
Les deux premiers articles de SAVFILE0 sont réservés aux checpoints successifs. Les valeurs de clés sont 000 et 001 en binaire sur trois octets. Le premier checkpoint est enregistré à la clé 000, le second à la clé 001, le troisième de nouveau à la clé 000 et ainsi de suite.

Justification:

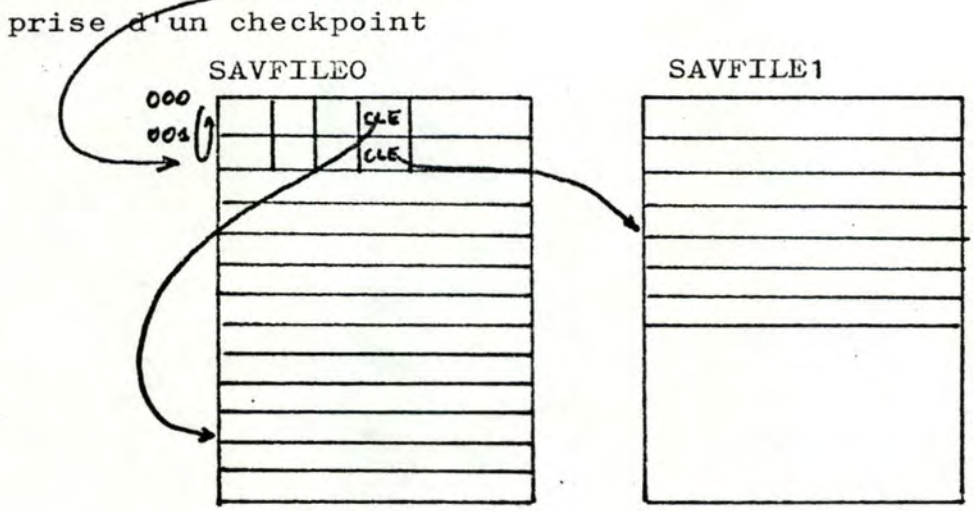
Si une panne survient au moment du checkpoint et si un seul article du fichier lui est réservé, le checkpoint précédent risque d'être détruit et le courant perdu si le contenu de la mémoire est endommagé.
Grâce au "flip-flop" le checkpoint précédent peut toujours être utilisé pour restaurer la base de données.

Le critère déterminant la prise d'un checkpoint est un nombre fixe de messages complètement traités. Le programme décriture du point de contrôle est appelé tous les 5 messages terminés.

zone gérée par le programme de sauvetage, appelée CPBUF (checkpoint buffer)



FILENAME = SAVFILE0 ou SAVFILE1



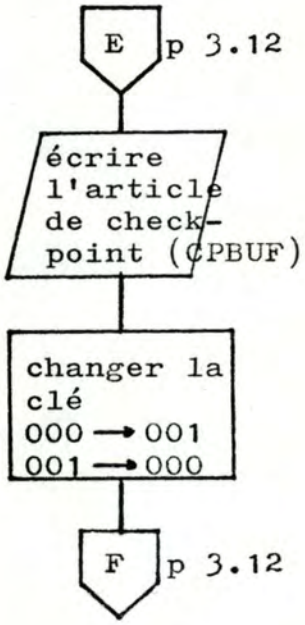


figure 3.9

33.26 Restauration. figure 3.10 a et b

Amener le checkpoint en mémoire. Lire les deux premiers enregistrements de SAVFILEO et comparer les numéros de séquence interne. Le numéro le plus grand donne le checkpoint le plus récent et c'est à partir de celui là que le traitement s'effectue.

Examiner le nom du fichier de sauvetage figurant dans le checkpoint pour déterminer le fichier contenant l'enregistrement sauvé correspondant.

Examiner la clé d'accès pour accéder directement à l'enregistrement.

Nous balayons les articles des fichiers de sauvetage jusqu'au dernier enregistrement sauvé avant la panne. La restauration doit se faire dans le sens dernier enregistrement sauvé vers l'enregistrement correspondant au checkpoint pour obtenir les fichiers applications dans l'état équivalent à celui du moment de la prise du checkpoint. Un même enregistrement mis à jour plusieurs fois se trouve sauvé un nombre égal de fois aux mises à jour. En restaurant dans le sens direct, il ne se truverait pas dans l'état qui était le sien au moment du checkpoint.

Exemple.

Article AA: contenu: 100

	← checkpoint	
màj1:	+10=110	sauvetage: 100
2:	+10=120	110
3:	- 5=115	120
4:	+10=125	115
	← panne	

restauration sens direct: l'article contient 115 ≠ 100 qui est le contenu correspondant au checkpoint.

restauration sens inverse: 115, 120, 110, 100 correct

Balayage.

A chaque lecture, comparer le numéro de séquence interne suivant (NS) avec le précédent (NP)

NS ≥ NP: NS devient NP, lire l'enregistrement suivant et recommencer la comparaison.

NS < NP: NP correspond au dernier article sauvé avant la pane et la restauration commence.

Restauration.

Pour chaque enregistrement sauvé:

a) comparer le numéro de séquence interne avec celui de l'enregistrement correspondant au checkpoint.
   En cas d'égalité, la restauretion est terminée et on relance l'exploitation.
b) sinon examiner le nom du fichier applications et l'adresse dans ce fichier.
c) écrire le contenu sauvé dans le fichier application adéquat.

A la fin de la restauration, les fichiers applications sont remis dans l'état équivalent au moment de la prise du dernier checkpoint.

Le programme de recouvrement (CREST) est appelé après une destruction logique des fichiers (cfr types de destruction p 2.1).

Voyons comment utiliser cette méthode en cas de destruction logique massive et de destruction physique des fichiers.

Destruction physique.

Pour des raisons de sécurité, nous sommes amenés à tirer périodiquement des copies ou dumps de la data base.

En cas de destruction physique de celle-ci, nous devons recharger le dernier dump et retraiter toutes les transactions entrées après la copie du dump. Le recouvrement peut être amélioré si nous enregistrons aussi dans

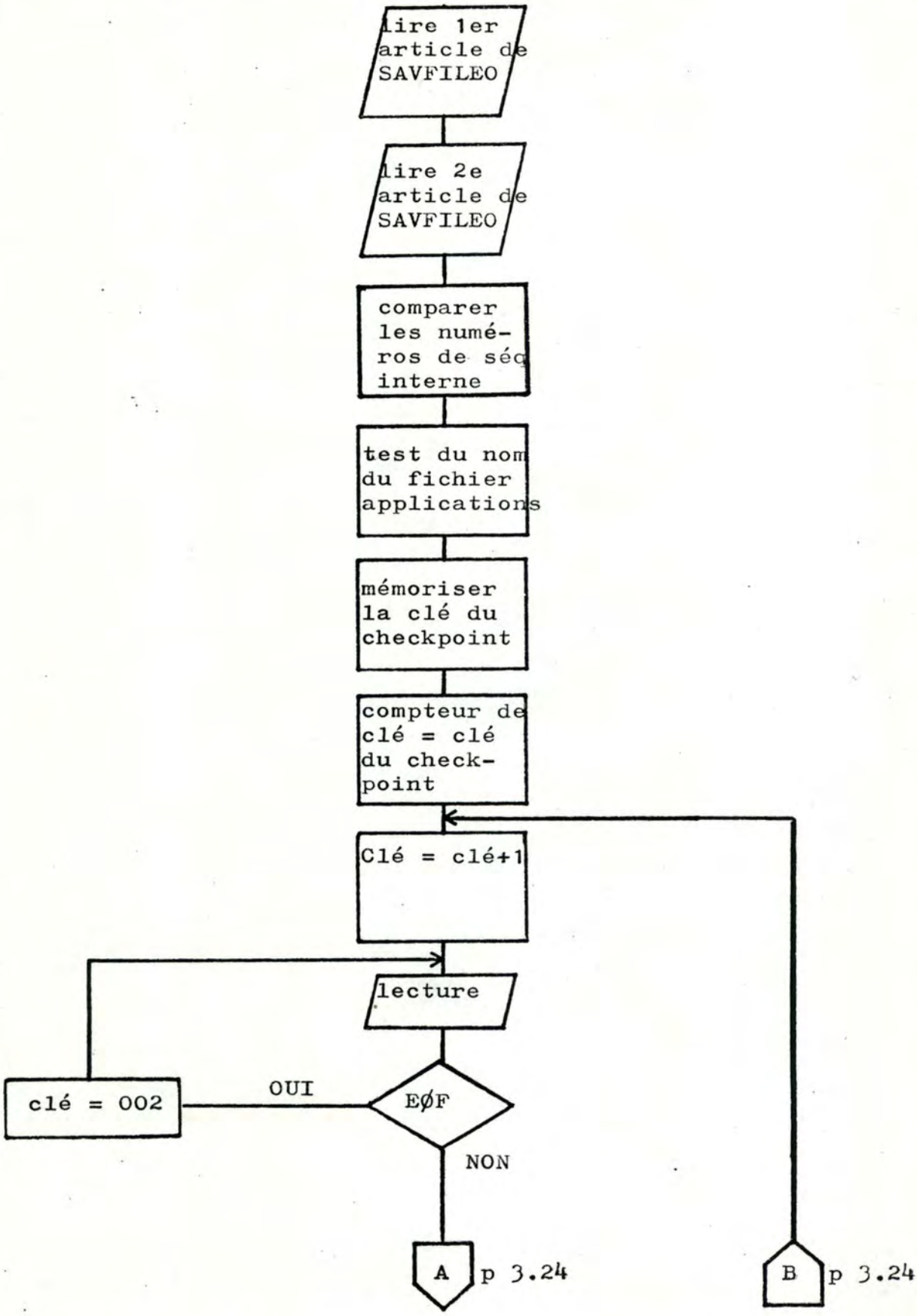

lire 1er article de SAVFILEO
lire 2e article de SAVFILEO
comparer les numéros de séq interne
test du nom du fichier applications
mémoriser la clé du checkpoint
compteur de clé = clé du check-point
Clé = clé+1
lecture
EØF
OUI
clé = 002
NON
A
p 3.24
B
p 3.24

figure 3.10 a

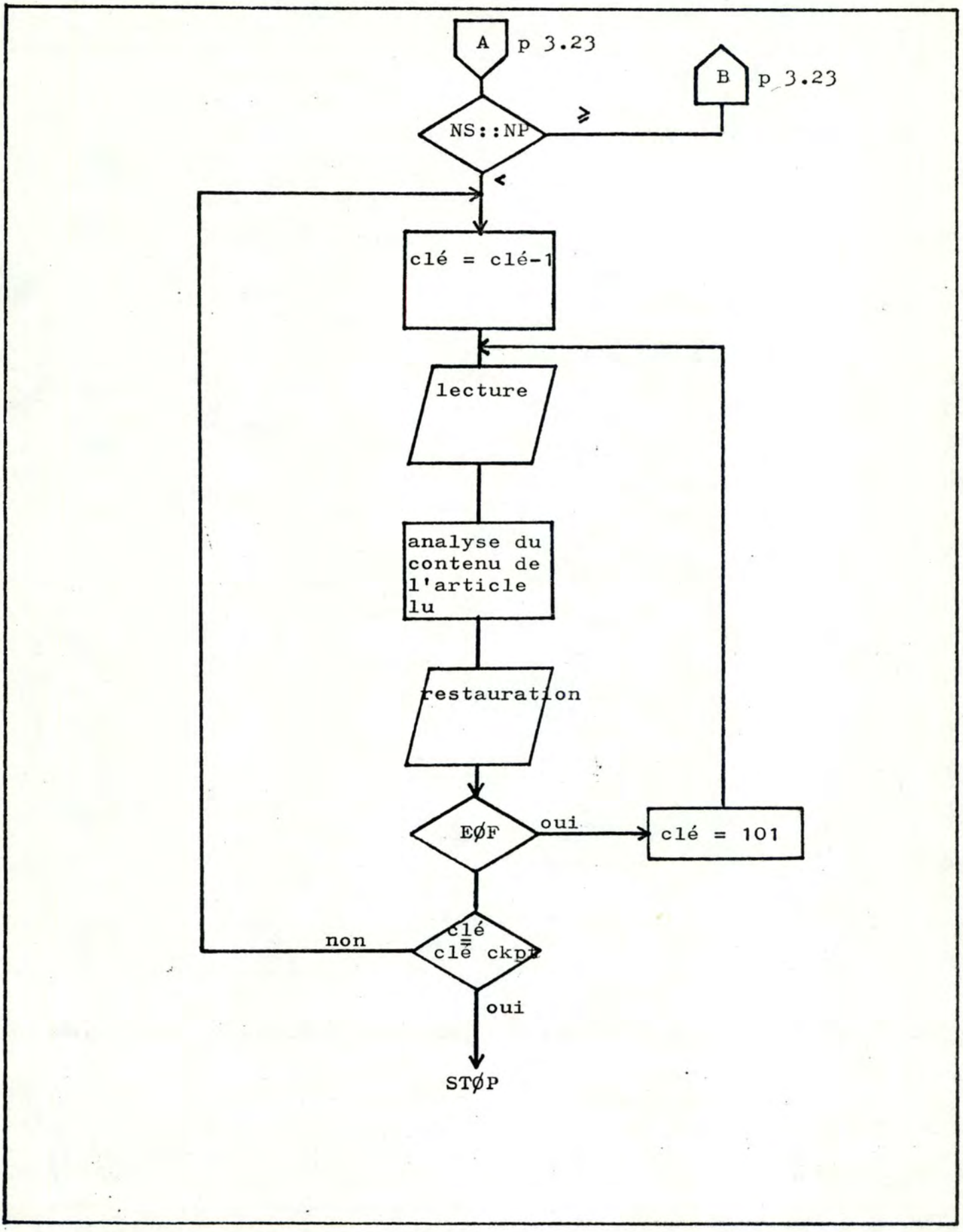

A
p 3.23
B
p 3.23
NS::NP
≥
<
clé = clé-1
lecture
analyse du contenu de l'article lu
restauration
EØF
oui
clé = 101
clé ≠ clé ckpt
non
oui
STØP

figure 3.10 b

les fichiers de sauvetage les copies après màj. Nous fusionnons alors le dump avec les fichiers de sauvetage (historiques subséquents au dump et courants au moment de la panne) en utilisant les copies après màj. La fusion s'arrête au checkpoint et on retraite les transactions à partir de ce point de contrôle.

Destruction logique massive.

Toujours avec l'exemple de l'erreur de programmation non détectée, nous ne pouvons pas compter sur les contenus des articles, avant mise à jour, dans les fichiers de sauvetage. Il faut recharger le dernier dump avant l'erreur et retraiter toutes les transactions subséquentes.

33.27 Redémarrage de l'exploitation.

a) retraiter automatiquement les messages entrés après celui correspondant au checkpoint.
Reprendre le dialogue avec les terminaux et autoriser l'envoi de nouveaux messages. La panne n'a pas, dans ce cas-ci d'incidencesur l'enchaînement des messages au niveau de chaque utilisateur. Les nouveaux messages sont enregistrés à la suite des autres dans le fichier de sauvetage des messages.

b) Ne pas recommencer le traitement des messages. Parcourir le fichier de sauvetage des messages pour renseigner les utilisateurs sur leur dernier input entièrement traité. A partir de là, ils doivent réintroduire leurs transactions et elles sont enregistrées dans le fichier de sauvetage des transactions immédiatement après celle correspondant au checkpoint. Le système leur affecte un numéro de séquence interne égal au numéro du checkpoint plus 1.

NB. Le redémarrage n'est pas développé dans notre système.

34 CONCLUSION

Parmi toutes les techniques de recouvrement de fichiers que nous avons étudiées, celle qu'il faudra choisir lors de l'élaboration d'un système dépendra avant tout des objectifs et des critères fixés par l'organisation utilisatrice (MTBF, MTTR, continuité du service,...).

Généralement, il ne faudra pas en retenir une en particulier, mais bien une combinaison de deux ou plusieurs d'entre elles afin de bénéficier des avantages de celles utilisées et pallier aux inconvénients de l'une par l'autre.

Toutefois, la technique du journaling ou de l'audit trail s'avère la plus efficace car elle permet de faire face à un plus grand nombre de situations. La combinaison d'une de ces méthodes avec la prise périodique de checkpoints en améliore encore le rendement.

Nous avons vu aussi que la sécurité exige d'exécuter des dumps de la base de données à intervalles réguliers, ceux-ci étant déterminés en fonction de la taille de la base de données et de la vitesse d'écriture des dumps.

LECTURE DES LISTINGS.

Nous trouvons en annexe un listing de l'ensemble des programmes et des résultats. Il se compose comme suit:

- listing des programmes CMAIN: programme principal
  CSECT2, CSECT3, CSECT4: programmes applications
  CSAUV: programme de sauvetage
  CKPT: programme de checkpoint
- listing du programme de restauration: CREST
- listing des résultats:
  . contenu des fichiers de sauvetages des articles avant mise à jour: SAVFILE0 et SAVFILE1
  . contenu des fichiers applications APFILE10 et APFILE11 après exécution des programmes CMAIN,... ..., CKPT

vis à vis: a) nous avons changé le contenu des fichiers applications pour montrer clairement l'effet de la restauration
b) contenu de APFILE10 et APFILE11 après exécution du programme de restauration CREST

NB. Voir le format des articles de SAVFILE0 et SAVFILE1 page 3.14 et des checkpoints page 3.20

Les deux premiers enregistrements de SAVFILE0 contiennent les checkpoints. Le numéro de séquence interne le plus grand est 0020. Le checkpoint le plus récent figure donc dans le premier article. (Dans les fichiers, le numéro de séquence interne est en binaire sur 4 octets; il est écrit sur le listing en 11 caractères)

L'enregistrement correspondant au checkpoint se trouve dans SAVFILE0 à la clé 051. Il s'agit donc du dernier article de la base de donnée manipulé par la dernière transaction terminée au moment de la prise du checkpoint.

A partir de la clé 051, nous balayons les fichiers de sauvetage jusqu'a rencontre d'un numéro de séquence interne inférieur à celui lu précédemment: 0008 est inférieur à 0021.

Nous restaurons ensuite la zone "CONTENU" dans les fichiers application adéquats et aux clés indiquées dans la zone "CLE", en sens inverse au balayage: d'où la restauration dans APFILE10: clés 007, 006, 005

APFILE11: clés 015 à 001

Un article mis à jour plusieurs fois aura, après restauration,un contenu équivalent au sien au moment de la prise du checkpoint.

ANNEXE 1

RECOUVREMENT EN I.D.S

L'I.D.S utilise le journaling comme méthode de recouvrement. Il enregistre toutes les transactions dans les pages du fichier journal, de même que toutes les actions dans les fichiers de la base de données. L'information de journalisation - obtenue à partir de chaque activité I.D.S en exécution - est stockée sur une bande magnétique, fournissant ainsi un fichier source utilisé pour la restauration de la data base à un état précédemment connu dans l'éventualité d'une détérioration de celle-ci.

Le fichier journal contient de multiples informations relatives à la procédure de journalisation et au fonctionnement de l'I.D.S lui-même.

Parmi celles-ci:

- copie des pages avant et après mise à jour.
- articles "images"des pages.

Les articles "images" sont de deux types selon que la copie est "before" ou "after" et écrits respectivement avant et après chaque modification. Ils contiennent des informations de synthèse sur le statut des pages du fichier journal.

Exemples:

- type de page: copie avant ou après
- nombre de lignes dans la page
- date et heure du démarrage de l'activité I.D.S
- nom du fichier I.D.S contenant la page

etc...

La bande magnétique de journalisation contient donc les copies des pages avant et après mise à jour.En cas de recouvrement, le journal est traité selon la figure a1.1

Séquence.

1) Etablir le critère de sélection pour obtenir les pages appropriées du fichier journal (relatives à la data base.) On se base pour cela sur diverses informations

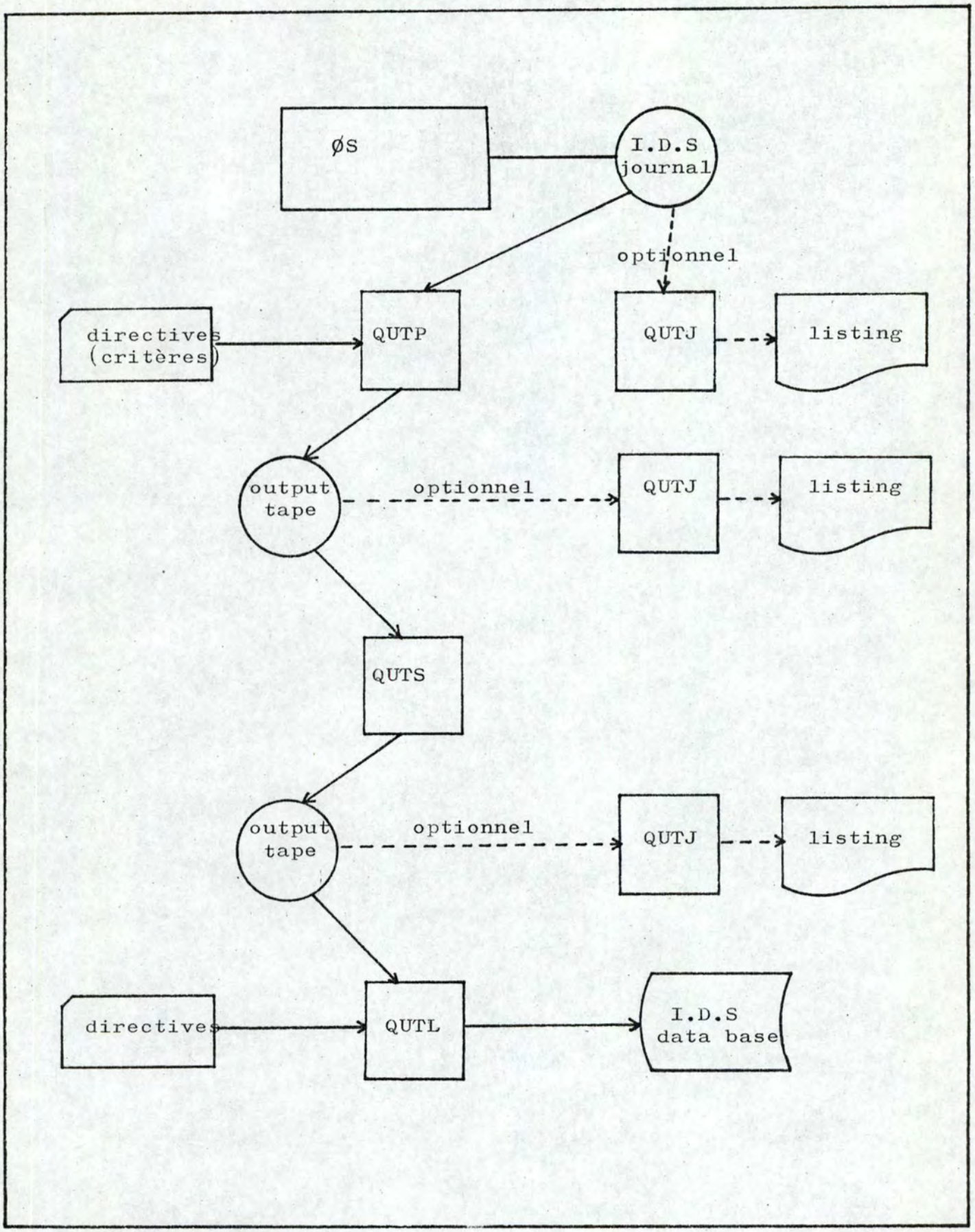

ØS
I.D.S
journal
optionnel
directives
(critères)
QUTP
QUTJ
listing
output
tape
optionnel
QUTJ
listing
QUTS
output
tape
optionnel
QUTJ
listing
directives
QUTL
I.D.S
data base

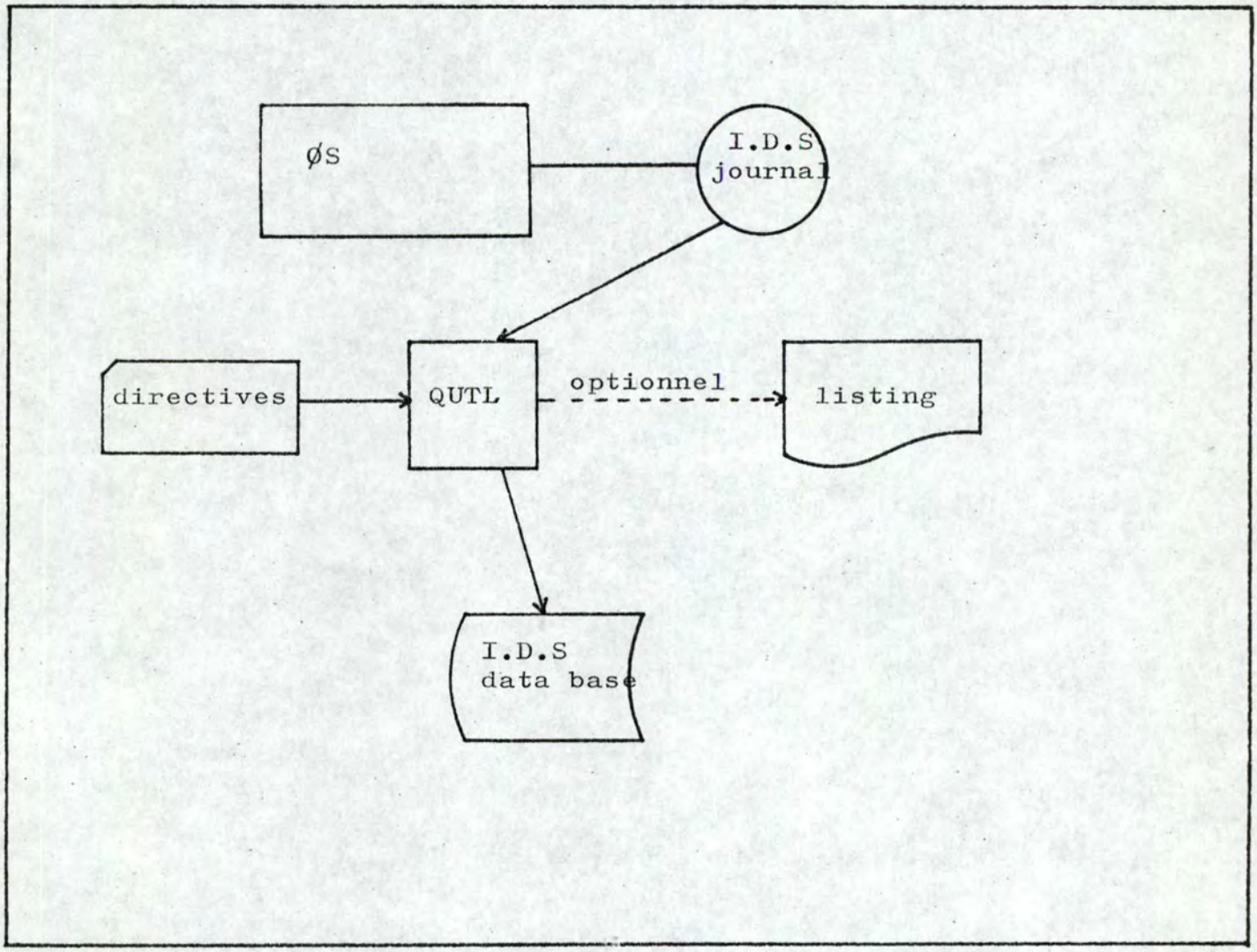
figure a1.1

telles que celles du rapport de l'activité de la data base.

2) La routine QUTP (utilitaire I.D.S) sélectionne les pages du journal tape.

3) La routine QUTS trie les pages sélectionnées pour éliminer celles portant le même numéro.
L'output trié consiste en la première copie avant ou la dernière copie after pour un numéro de page donné, à recharger dans la data base.

4) La routine QUTL recharge l'output dans la portion appropriée de la data base. Elle peut être aussi utilisée pour charger les pages directement du journal. (figure a1.2



ANNEXE 2

RECOUVREMENT EN I.M.S

La philosophie générale du recouvrement en I.M.S est d'enregistrer le statut du système (programmes, fichiers etc...) à un moment donné (checkpoint) et d'enregistrer tous les changements de statut à partir de ce point jusqu' au au prochain checkpoint ou jusqu'à la panne.

L'enregistrement des changements de statuts est appelé "logging" en I.M.S; c'est une forme de journaling.

En cas de panne, le recouvrement consiste à restaurer le statut de la base de données au moment du checkpoint et à recréer l'activité de changement du statut.

Recouvrement de la data base.

Principe:

Constituer une copie back-up à un moment donné et garder un enregistrement de tous les changements survenus dans la base de données.

a) copie de la data base.

Un programme utilitaire fournit à grande vitesse une copie back-up de la data base. Celle-ci est divisée en ensembles de données (data sets). Tous les ensembles sont copiés en une fois pour plus de facilité et l'utilisateur peut de mander l'exécution de deux dumps en parallèle. En cas d'erreur d'entrée/sortie dans l'un d'eux, le programme continue et achève l'autre.

La copie est créée sur bande magnétique. Le premier enregistrement de celle-ci est appelé"dump header", il contient des informations d'identification des ensembles de données (data sets): date, heure,...

La fréquence des dumps est dépendante des exigences des utilisateurs en durée de recouvrement.

b) accumulation des changements dans la data base.

Le programme spécialisé à cette tâche dispose de l'ancien fichier accumulation des changements survenus dans la data base et du fichier journal (log tape).

Il exécute la séquence suivante:

- lecture des enregistrements du journal. Comme celui-ci contient des informations sur tout le système, le programme ne retient que celles concernant les chengements intervenus dans la base de donées.
- fusion de l'ancien fichier accumulation avec les nouveaux changements.

  Le résultat est un nouveau fichier accumulation contenant la valeur globale de tous les changements survenus dans la data base à partir du moment de la copie back-up.

Le volume des informations à traiter en cas de recouvrement est considérablement réduit et celui-ci est donc plus performant.

Le fichier accumulation sert aussi à sélectionner les articles appartenant aux parties les plus importantes de la data base pour les grouper en un autre fichier accumulation utilisé en cas de recouvrement extrêmement urgent des seules parties critiques de la base de données.

c) recouvrement.

Le recouvrement s'effectue en deux phases:

- fusion de la dernière copie back-up de la base de données avec le fichier accumulation des changements. La data base est alors entièrement ou partiellement restaurée.

  Partiellement restaurée signifie qu'il existe au moment de la panne un ou plusieurs fichiers journaux non encore fusionnés avec le fichier accumulation. Dans ce cas:
- mise à jour de la data base à partir de ces journaux.

Remarque.

Il existe en I.M.S plusieurs formes de checkpoints:

- checkpoints normaux (simple checkpoints): ce sont les points de contrôle courants, donnant une référence, un point de synchronisation pour le redémarrage du système.
  Plus leur fréquence est petite, plus il y a d'enregistrements sur les journaux et plus le redémarrage urgent est long (redémarrage à l'endroit de la panne).
  Une fréquence trop élevée dégrade les performances du système.
- checkpoints d'arrêt (termination checkpoints): ce sont des points de contrôle pris lors d'un arrêt de l'activité I.M.S. Ils permettent l'achèvement des transactions en cours mais en interdisent de nouvelles. La file d'attente des transactions est bloquée, de même que la data base et le fichier journal est fermé.

Si le dernier journal utilisé pour le recouvrement n'a pas été fermé par un "termination checkpoint", la data base n'est pas correctement restaurée et les changements relatifs à ce journal en sont effacés (data base backout).

D'une manière générale, le "backout" consiste à supprimer, après exécution d'un programme, les changements effectués par ce dernier. Comme il s'agit de changements de la data base, un fichier journal de backout (backout log tape) est créé et il doit être utilisé lors d'une procédure de recouvrement. La base de données après le backout ne se trouve pas nécessairement dans l'état équivalent à celui précédent l'exécution du programme.

Exemple:

Une zone libérée par un delete peut déjà être utilisée par un autre programme avant la fin du backout.

Quand elle est appelée pour un recouvrement, la procédure "data base backout" efface donc les changements engendrés par les messages en activité au moment de la panne.

Sécurité des journaux.

L'importance des journaux est essentielle pour assurer l'intégrité de la base de données. Leur sécurité est donc l'objet d'une attention particulière.

L'écriture dans les journaux étant bloquée et bufferisée, les changements peuvent avoir eu lieu dans la data base avant enregistrement de ceux-ci dans les journaux.

Quand l'I.M.S se termine anormalement et que l'ØS reste actif, une routine force l'écriture du contenu des buffers dans les journaux et ferme ceux-ci. L'intervention de l'opérateur est cependant quelques fois requise.

Le problème est de retrouver les données n'ayant pas pu être écrites sur le "log tape" avant la panne et de les ajouter aux informations déjà écrite dans les journaux. On se base pour cela sur l'existence d'un dump du contenu de la mémoire au moment de la panne. Si l'ØS reste actif, il effectue le dump, sinon il faut recourir à une procédure spéciale pour l'exécuter.

Une fois l'ØS relancé, les buffers non sauvés sont localisés et leur contenu est écrit dans les fichiers journaux.

BIBLIOGRAPHIE.

++++++++++++++++++++++++

- YOURDON Edward

  Design of on-line computer systems.

- MARTIN James

  System analysis for data transmission.

  Design of real-time computer systems.

  Programming real-time computer systems.

  Security, accuracy and privacy in computer systems.

- VAN TASSEL Dennis

  Computer security managment.

- WINDAL Jean-Pierre

  Cours de télétraitement.

- Documentation I.D.S

- Documentation I.M.S